Coventry University

DOCTOR OF PHILOSOPHY

Cloud computing based adaptive traffic control and management

Jaworski, Pawel

Award date:
2013

Awarding institution:
Coventry University

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of this thesis for personal non-commercial research or study
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission from the copyright holder(s)
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 29. May. 2021

Cloud Computing based Adaptive Traffic Control and Management

Paweł Jaworski

A thesis submitted in partial fulfilment of the University's requirements for the degree of Doctor of Philosophy.

September 2013

Control Theory and Applications Centre

Coventry University

In collaboration with MIRA Ltd.

Abstract

Recent years have shown a growing concern over increasing traffic volume worldwide. The insufficient road capacity and the resulting congestions have become major problems in many urban areas. Congestions negatively impact the economy, the environment and the health of the population as well as the drivers satisfaction.

Current solutions to this topical and timely problem rely on the exploitation of Intelligent Transportation Systems (ITS) technologies.

ITS urban traffic management involves the collection and processing of a large amount of geographically distributed information to control distributed infrastructure and individual vehicles.

The distributed nature of the problem prompted the development of a novel, scalable ITS-Cloud platform. The ITS-Cloud organises the processing and manages distributed data sources to provide traffic management methods with more accurate information about the state of the traffic. A new approach to service allocation, derived from the existing cloud and grid computing approaches, was created to address the unique needs of ITS traffic management. The ITS-Cloud hosts the collection of software services that form the Cloud based Traffic Management System (CTMS). CTMS combines intersection control algorithms with intersection approach advices to the vehicles and dynamic routing.

The CTMS contains a novel Two-Step traffic management method that relies on the ITS-Cloud to deliver a detailed traffic simulation image and integrates an adaptive intersection control algorithm with a microscopic prediction mechanism. It is the first method able to perform simultaneous adaptive intersection control and intersection approach optimization. The Two-Step method builds on a novel pressure based adaptive intersection control algorithm as well as two new traffic prediction schemes.

The developed traffic management system was evaluated using a new microscopic traffic simulation tool tightly integrated with the ITS-Cloud. The novel traffic management approaches were shown to outperform benchmark methods for a realistic range of traffic conditions and road network configurations. Unique to the work was the investigation of interactions between ITS components.

Acknowledgements

Foremost, I would like to express my gratitude towards my Director of Studies Dr. Olivier Haas for his guidance and support in realisation of this project. I am grateful to Prof. Keith Burnham and Dr Anthony Baxendale for giving me the opportunity to undertake this project, their valuable input and continuous support.

I would also like to thank my colleagues at MIRA Ltd. for all the discussions we had that often set me on the right path towards solving many problems. I would like to acknowledge the input of my industrial supervisor Tim Edwards for his expertise and frequent exchange of ideas.

I would like to express my gratitude to the Coventry City Council for providing access to the live traffic data from the city traffic management system.

To my family and friends, I am thankful for their encouragement and support.

Last but not least, I would like to thank Agnieszka Beza for her love and understanding that was invaluable in my pursue of this project.

Contents

1 Introduction 1

1.1 Introduction and research motivation 1

1.2 The aim and objectives 3

1.2.1 The aim of the project 3

1.2.2 Objectives 4

1.3 Methodology 4

1.4 Deliverables and original contributions 6

1.5 Outline of approach 9

2 Literature background 11

2.1 Traffic management criteria, constraints and objectives 12

2.2 Traffic management and control 13

2.2.1 Traffic light based traffic control 14

2.2.2 Vehicle actuated traffic control 16

2.2.3 Vehicle detection 18

2.3 Traffic modelling and simulation 19

2.3.1 Macroscopic traffic models and simulators 20

2.3.2 Microscopic traffic modelling and simulation 21

2.4 Vehicle control methods 23

2.4.1 Longitudinal control 24

2.4.2 Lateral control 26
2.5 Communication methods in the vehicular environment 27
2.5.1 Networking in vehicular environment 28
2.6 Distributed processing and its applications 29
2.7 Discussion 32
2.8 Conclusion 33

3 Traffic Management Methods 34
3.1 Benchmark algorithms 35
3.1.1 Algorithm 1: Fixed cycles intersection control algorithm . . . 35
3.1.2 Algorithm 2: ILC intersection control algorithm 38
3.2 ITS Pressure intersection control algorithm 40
3.3 The Two-Step Traffic Optimisation Method 43
3.4 Intersection Approach Trajectory Optimisation 45
3.5 The Two-Step process illustration 48
3.6 Dynamic routing 50
3.7 Conclusions 53

4 The *ITS-Cloud* platform 55
4.1 The ITS-Cloud platform 56
4.2 Service types 57
4.3 Multi-dynamic service allocation 59
4.4 Service discovery and deployment 60
4.4.1 Service discovery 61
4.4.2 Service deployment 63
4.4.3 Service migration and duplication 64
4.5 Reliability, fault tolerance and redundancy 65
4.6 The cloud interface and user interaction 66

4.7 Messaging . . . 67
4.8 Conclusions . . . 70

5 The Cloud based Traffic Management System (CTMS) 72

5.1 CTMS overview . . . 73
5.2 Approach to traffic management . . . 77
5.3 Creation of a situation image . . . 78
5.4 Meso Scale Prediction Service . . . 81
5.5 Micro Scale Prediction Service . . . 86
5.6 Intersection Control . . . 87
5.7 Intersection Approach Trajectory Optimisation . . . 92
5.8 Failure modes . . . 94
5.9 Conclusions . . . 96

6 The traffic simulator 97

6.1 Goals of the simulator . . . 97
6.2 Simulator design . . . 98
6.3 Vehicle simulation . . . 101
  6.3.1 Vehicle dynamics - longitudinal behaviour . . . 102
  6.3.2 Vehicle dynamics - longitudinal model tuning . . . 105
  6.3.3 Vehicle speed control . . . 110
  6.3.4 Lane changing - lateral behaviour . . . 111
  6.3.5 Vehicle following (Platooning) . . . 115
  6.3.6 Cooperative platooning . . . 117
  6.3.7 Off-board speed advice . . . 118
  6.3.8 Driver modelling . . . 119
6.4 Simulation of wireless communication . . . 122
6.5 Road network simulation . . . 123

6.5.1 Road simulation . . . 124
6.5.2 Intersection simulation . . . 124
6.5.3 Infrastructure simulation . . . 126
6.6 Defining simulation scenarios . . . 127
6.7 Gathering simulation data . . . 128
6.8 ITS-Cloud integration . . . 129
6.9 Conclusions . . . 131

7 Simulation studies 134

7.1 Simulation set-up and scenarios . . . 135
7.1.1 ITS-Cloud platform . . . 136
7.1.2 Urban areas . . . 136
7.1.3 Long corridor . . . 139
7.1.4 Single intersection scenario . . . 139
7.1.5 Coventry scenario . . . 140
7.1.6 Arterial road scenario . . . 141
7.1.7 Grid City scenario . . . 142
7.2 Result quantification . . . 144
7.3 Cooperative platooning evaluation . . . 145
7.3.1 Experiment 1 - freeway . . . 146
7.3.2 Experiment 2 - Intersection clear time . . . 148
7.3.3 Experiment 3 - Impact of traffic lights . . . 150
7.3.4 Experiment 4 - Impact on journey times in urban environment 153
7.4 Intersection Approach Trajectory Optimisation . . . 155
7.5 Dynamic routing . . . 158
7.6 The Micro Scale Prediction Service . . . 160
7.7 The Meso Scale Prediction Service . . . 163
7.8 Parameter sensitivity . . . 166

7.8.1 Experiment 1 - ICA sampling rate 166

7.8.2 Experiment 2 - ITSP speed weight 167

7.8.3 Experiment 3 - the time headway 168

7.8.4 Experiment 4 - Speed limit 169

7.9 Cloud system performance 170

7.9.1 Impact of data processing latency 171

7.9.2 Impact of network latency 173

7.10 Evaluation of complex traffic management solutions 176

7.10.1 Coventry scenario analysis 177

7.10.2 Arterial Road scenario analysis 181

7.10.3 Grid City scenario analysis 184

7.10.4 Summary 187

7.11 Conclusions and findings summary 189

8 Conclusions and Future Work **193**

8.1 Conclusions 193

8.2 Deployment plan 198

8.2.1 MIRA deployment 198

8.2.2 Coventry city deployment 199

8.3 Limitations and future work 199

References **219**

Appendices **220**

A Traffic simulator application **221**

A.1 Scenario configuration 221

A.2 Modes of operation 226

A.3 Batch mode 226

A.4 Graphical user interface mode 227
A.4.1 Traffic simulation view 229
A.4.2 Traffic simulation charts view 230
A.4.3 Vehicle administration view 231
A.4.4 Sensor view 231
A.4.5 Options view 232

B ITS-Cloud software 234

B.1 Deployment of ITS-Cloud registry 234
B.2 Resource deployment 235

C Paper - Cloud Computing Concept for Intelligent Transportation Systems 236

D Paper - Distributed Traffic Flow Optimisation and Control for Intelligent Transportation Systems 243

E Paper - Microscopic Traffic Simulation Tool for Intelligent Transportation Systems 251

F Paper - A Network Assisted Vehicle for ADAS and ITS testing 258

G Paper - Autonomous longitudinal control for a Network Assisted Vehicle 264

List of Figures

1.1 PhD project Gantt diagram 5

3.1 Traffic cycle and stages 36
3.2 Intersection configuration example 37
3.3 IATO decision diagram 47
3.4 Two-Step traffic management diagram 49
3.5 Road network graph representation 51
3.6 Dynamic Routing graph search result example 53

4.1 Cloud and grid service models 57
4.2 Components and services of the ITS-Cloud 58
4.3 Multi-dynamic service allocation - first client 59
4.4 Multi-dynamic service allocation - following clients 60
4.5 ITS-Cloud layer model 66
4.6 Class diagram of the message hierarchy in the ITS-Cloud. 69

5.1 Information flow within the CTMS. 74
5.2 Traffic management organisation in CTMS 77
5.3 Construction of a situation image using data from various sources. . . 80
5.4 MeSPS notifications 82
5.5 MeSPS prediction validation measurement 84

5.6 Intersection control process 88
5.7 Intersection Control Service 89
5.8 Intersection management process using the Two-Step method. 91
5.9 Geographical addressing in CTMS 93

6.1 Simulator component class diagram 99
6.2 Simulation cycle diagram 100
6.3 Vehicle component class diagram 103
6.4 The vehicle model 105
6.5 Vehicle speed and acceleration - tuning data set 107
6.6 Vehicle speed and acceleration - validation data set 108
6.7 Vehicle model parameter range 109
6.8 Low level vehicle controller 110
6.9 Lane change decision diagram 112
6.10 Lane changing illustration 115
6.11 Measured DSRC latency 123
6.12 Example lane configuration 124
6.13 Cross type intersection traffic flow compatibility 125
6.14 Linking two simulator instances 130

7.1 Traffic flow measured in Coventry 138
7.2 The Coventry scenario topology. 140
7.3 The arterial road scenario topology. 142
7.4 The Grid City scenario topology. 143
7.5 ACC Platoon - separation 147
7.6 CACC Platoon - separation 147
7.7 Intersection clear times of ACC and CACC platoons 149
7.8 Kinetic energy on intersection approach - ACC 150

7.9 Kinetic energy on intersection approach - CACC 151
7.10 Accumulated kinetic energy on intersection approach 152
7.11 Influence of CACC on journey time 154
7.12 IATO subsystem performance 156
7.13 Dynamic Routing subsystem performance 159
7.14 Accuracy of MiSPS prediction 161
7.15 MeSPS traffic flow prediction 165
7.16 Performance of local ITSP and cloud based ITSP 172
7.17 Measured latency between *mira.co.uk* and selected destinations. . . . 174
7.18 Impact of network quality on traffic management 175
7.19 Coventry scenario, high traffic density 178
7.20 Coventry scenario, low traffic density 179
7.21 Arterial scenario, high traffic density, high speed 182
7.22 Grid City scenario, high traffic density, high speed 185
7.23 Grid City scenario, high traffic density, low speed 186

A.1 Main traffic stimulation view 227
A.2 Traffic simulation view 229
A.3 Traffic simulation - charts 230
A.4 Vehicle Administration 231
A.5 Traffic simulation sensors 232
A.6 Traffic simulation options 233

List of Tables

2.1 Comparison of cloud and grid systems 31

5.1 System failure modes 94
5.2 Service failure consequences 95

6.1 Model tuning results 107
6.2 Mean Square Errors (MSE) 108
6.3 Intersection flow compatibility truth table 125

7.1 ICA features and compatibility with the CTMS system components . 134
7.2 Test bed configuration 136
7.3 Impact of ICA sampling rate 166
7.4 Impact of ITSP speed weighting β_V 168
7.5 Impact of time headway τ_{ACC} 169
7.6 Speed limit impact on throughput [veh/h] 170
7.7 Journey time T_J summary 187
7.8 Energy consumption E_C summary 188

Abbreviations

ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance Systems
AICC	Autonomous Intelligent Cruise Control
AMS	Area Management Service
ATLC	Adaptive Traffic Light Control
CACC	Cooperative Adaptive Cruise Control
CTMS	Cloud based Traffic Management System
DSRC	Dedicated Short Range Communications
DVLA	Driver and Vehicle Licensing Agency in the UK
FC	Fixed Cycles ICA
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HMI	Human-Machine Interface
HUD	Head Unit Display
IATO	Intersection Approach Trajectory Optimisation
ICA	Intersection Control Algorithm
ICS	Intersection Control Service

ILC	Induction Loop vehicle Count ICA
ITS	Intelligent Transportation Systems
$ITS-vehicle$	A vehicle capable of V2x communication
$MeSPS$	Meso Scale traffic Prediction Service
$MiSPS$	Micro Scale traffic Prediction Service
NAV	Network Assisted Vehicle
RCS	Radio Communication Service
SDS	Service Discovery System
$SOAP$	Simple Object Access Protocol
SS	Sensor Service
TMS	Traffic Management System
$V2I$	Vehicle-to-Infrastructure wireless communication
$V2V$	Vehicle-to-Vehicle wireless communication
$V2x$	Vehicle to other devices wireless communication
$VANET$	Vehicular ad-hoc Network
$VATC$	Vehicle Actuated Traffic Control

Variables

a_t	Vehicle acceleration at time t
$C_{i,j}$	Class (type) of vehicle i on lane j
CP	Cycle plan, a list of stages and their durations
d_s	Distance between the vehicle and the stop line
$d_{v,T}$	Distance covered in T_{switch} while travelling at $V_{j,max}$
d_{sensor}	distance between the sensor and the stop line
d_t	Distance travelled by the vehicle in time t
D_j	Amount of outflows form lane j
$D_{A,B}$	Distance between intersections A and B
δ_x	ITS sensor mean location measurement error
δ_V	ITS sensor mean speed measurement error
e_t	Estimated energy consumption at time t
$f_{i\|j}$	Flow direction j associated with signalling stage i
g_i	Signalling stage i
H_p	Prediction horizon in the Two-Step method
H_a	Advice horizon in the Two-Step method
ι_i	Lane dissatisfaction parameter
$\tilde{N}_j$	Estimated amount of vehicles to choose outflow j
$N_{id,s}$	Amount of vehicles released from stage s of intersection id
$\bar{N}_{id,s}$	The amount of vehicles estimated to arrive form stage s of intersection id
p_t	Position of a vehicle with respect to the beginning of the lane
$P_{out\|j}$	Probability of vehicle choosing outflow direction j
$\pi_{i,j}$	ITS pressure index of vehicle i on lane j

Π_s	Total ITS pressure associated with signalling stage s
R_s	ITS-Cloud resource suitability index
R_{count}	Amount of services deployed on resource
R_{CPU}	Amount of CPU cores available on resource
s_p	Separation distance between a vehicle and its predecessor
$s_p^{desired}$	Desired separation distance between a vehicle and its predecessor
Δs_p	Separation error - difference between s_p and $s_p^{desired}$
$s_{p,max}$	Preceding vehicle detection range
s_f	Separation distance between a vehicle and its follower
s_0	Minimal separation between vehicles
S_j	Situation image on lane j
$\hat{S}_j$	Predicted situation image on lane j
$t_{g,i}$	Duration of signalling stage i
t_s	Time until signalling stage changes
$t_{id,s}$	Time passed since intersection id activated stage s
t_{start}	MeSPS tuning - validation measurement start time
t_m	MeSPS tuning - validation measurement duration
Δt	Simulator sample length
T_j^Q	Estimated time required to clear queue on lane j
$T_{s,min}$	Minimal signalling stage duration
T_{CP}	Length of a signalling cycle
$T_{g,i}$	Start time of stage i relative to the begging of the cycle
T_{switch}	Time required to switch traffic light stages
T_p	Processing time
$T_{thr,brk}$	Throttle to brakes switching time

$T_{brk,thr}$	Brakes to throttle switching time
T_{LM}	Link-Model traversal time
$\theta_{id,s}$	Weighting parameter for stage s of intersection id
V	Current vehicle speed
V_t	Vehicle speed at time t
$V_{i,j}$	Speed of vehicle i on lane j
V_{lead}	Platoon leader speed
$V_{desired}$	The speed vehicle is aiming to achieve
$V_{lead}^{desired}$	The speed platoon leader is aiming to achieve
$x_{i,j}$	Location of vehicle i on lane j with respect to begging of the lane
V_p	Speed of the preceding vehicle
ΔV_p	Speed difference between a vehicle and its predecessor
$\tilde{V}_p$	Speed of potential preceding vehicle (when changing lanes)
V_f	Speed of the following vehicle
$\tilde{V}_f$	Speed of potential following vehicle (when changing lanes)
$V_{j,max}$	Speed limit on lane j
$(VC)_{j\|t}$	Mean vehicle count on link j calculated using last t time units of traffic data
$(VF)_{j\|t}$	Mean vehicle flow on link j calculated using last t time units of traffic data
W_j	Cost of traversing road j
q_j	Amount of queued vehicles on lane j
$x_{lead}(t_{id,s})$	Estimated location of platoon lead vehicle released at $t_{id,s}$ by stage s of intersection id
ξ	Driver aggravation coefficient

Chapter 1

Introduction

This chapter provides an introduction to the research carried out in this project. The main aim is identified leading to the definition of the objectives. Novelties and deliverables are then presented. The chapter finishes with an outline of the subsequent chapters.

1.1 Introduction and research motivation

According to the DVLA [1] the amount of vehicles in the UK has increased on average by 2.4% per year between 1996 and 2007 before slowing down to 0.5% in the 2008-2009 period due to the financial crisis. The following years have seen some recovery, with the amount of vehicles growing by 0.9% every year between 2010 and 2012. The growth rate is even greater in developing countries, where a rapid growth of the amount of vehicles is often accompanied by an underdeveloped road infrastructure [2]. The congestion caused by the insufficient road throughput was estimated to cost the UK economy over 4 billion pounds per year [3], mainly due to lost revenue and wasted fuel [4].

Dealing with traffic congestion has been made a high priority and deemed necessary for further social and economic development of any country [5]. Congestion can be reduced by increasing road capacity, which in turn requires significant investments in new road infrastructure. Such investments are usually very costly, time consuming and not always feasible owing to a lack of necessary space.

Intelligent Transportation Systems (ITSs) aim to improve existing road networks capacity, reduce travel times, fuel consumption, increase safety of all traffic participants and deliver traffic relevant information to the drivers [6]. Many ITS technologies rely on cooperation between vehicles and the infrastructure that is enabled by wireless communication.

An ITS aims to achieve those goals using a variety of technologies ranging from intelligent adaptive intersection control to advising the vehicles on optimal behaviour. Currently ITS research favours infrastructure based adaptive traffic control strategies in urban environments [7, 8, 9]. Besides adaptive traffic light control the infrastructure based ITS includes systems such as intersection approach optimisation [10, 11], that knowing the future state of the traffic lights can advise the vehicles on how should they approach, or dynamic vehicle routing [12, 13] that can suggest alternative routes to vehicles.

Self organising ad-hoc wireless networks are an enabling technology for infrastructure-less traffic management on motorways and the suburban environment [14, 15, 16]. Cooperative platooning are also referred to as cooperative adaptive cruise control (CACC) aims to improve road throughout and increase safety by adjusting the speed of the vehicles in a cooperative manner.

Adaptive traffic control strategies manage traffic flow in reaction to current traffic conditions, therefore use of various sensing techniques is required to provide such information. Increasing scale and coverage of ITS, especially in urban areas, requires collection and analysis of large amounts of geographically distributed data [17]. Such information may also be required by other ITS applications, different than traffic control, therefore a way of managing large amount of geographically distributed data and sharing it with multiple ITS applications is required.

This research project recognised both the necessity of managing such a large amount of distributed data and the need to improve the way urban traffic is managed. To aid with large scale data and processing management, a distributed processing platform called ITS-Cloud, was designed using Cloud computing principles and implemented as part of this project. Furthermore several ITS traffic management applications were developed using the ITS-Cloud platform. Those applications were designed to control various aspects of traffic flow and by cooperating with each other improve road utilisation, reduce journey times and energy consumption. Such a collection of cooperative distributed applications has been placed under a collective name of the Cloud based Traffic Management System (CTMS).

1.2 The aim and objectives

1.2.1 The aim of the project

The aim of this research was to simultaneously improve traffic flow and driver satisfaction in urban and suburban areas by reducing average journey times, energy consumption and carbon emissions as well as reducing the amount of stops required to be made by the driver when traversing an urban road network. To achieve this aim a number of objectives, identified in the following section, were realised leading to the development of a robust, distributed and scalable platform for hosting ITS applications onto which was deployed new traffic simulation and management tools.

1.2.2 Objectives

The key objectives, crucial to achieving the aim of the project, were identified as follows:

- Identify the criteria used to evaluate traffic management schemes.
- Investigate existing traffic management schemes and determine the most effective or most promising approaches. Examine their advantages and disadvantages and determine how they can be improved.
- Design an improved traffic management scheme to address the identified weaknesses of existing methods.
- Design and develop the ITS-Cloud, a robust, distributed and scalable execution platform on which the traffic management applications can be deployed.
- Design and develop a simulation tool that is integrated with the ITS-Cloud and is capable of performing accurate traffic simulations and cooperate with the developed traffic management applications.
- Design and develop all the system components and applications of the Cloud based Traffic Management System (CTMS).
- Validate the new traffic management methods using the traffic simulator.

1.3 Methodology

The realisation of this research project has been divided into three phases:

- Review of the current state of art and refinement of the project goals and deliverables.
- Design and implementation of intersection algorithms, ITS-Cloud platform, the Cloud based Traffic Management system and the traffic simulator.
- Analysis phase focused on performing simulation studies using the developed tools and techniques. Results were analysed and reported in this document.

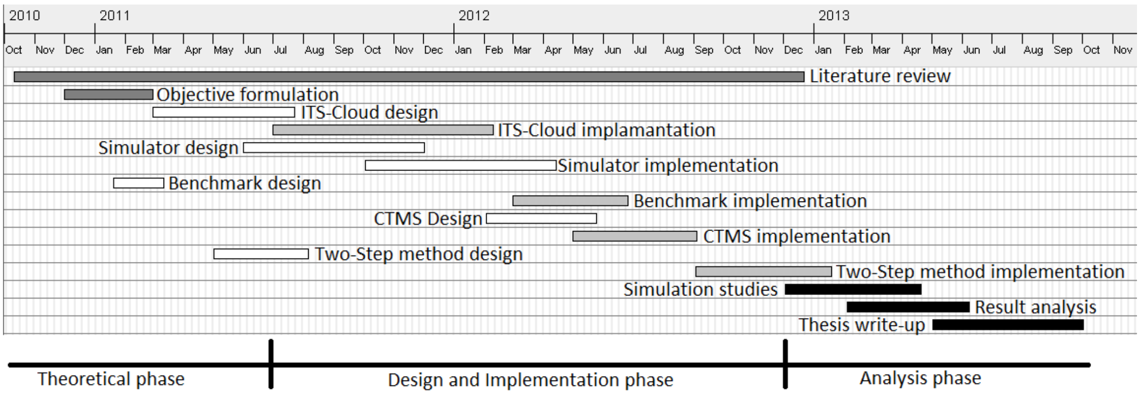


Figure 1.1: PhD project Gantt diagram

Figure 1.1 shows the Gantt chart representation of the tasks and processes involved in this project. The design processes are coloured white, implementation tasks are shown in grey and the tasks dealing with result collection, analysis and reporting are black. The design processes and implementation tasks overlap each other as it was often necessary to make design changes owing to encountered implementation issues. Additionally the development processed of the simulation tool and the ITS-Cloud was carried out simultaneously for most of the time. It is worth noticing that the literature review process was active through the most of the project in order to keep track of the constantly evolving state of art.

The software components were designed using an object oriented programming approach. The design diagrams were created using the *Unified Modelling Language* (UML). The resulting UML diagrams were then reviewed before being implemented using the *Java-SE* platform. Unit testing was carried out using the *Jtest*, a *Java* unit testing tool kit. The vehicle components were initially developed and validated using the *MATLAB/Simulink* environment before being implemented in the *Java* based traffic simulator.

1.4 Deliverables and original contributions

This research project introduced several novelties and proposed novel applications for existing technologies. The novelties are as follows:

- **Joint traffic light and vehicle speed optimisation**

  A novel traffic management method referred to as 'Two-Step traffic optimisation' within this work. It combines adaptive traffic light control with the capability to instruct the appropriately equipped vehicles on how to approach the intersection with aim to reduce journey times and conserve energy. This has been identified in the literature as an important capability that is currently missing and as such limits the application of promising traffic management techniques such as intersection approach optimisation. It is therefore the most important novelty of this work.
- **Application of distributed processing principles in the traffic management field**

  In order to achieve scalability and robustness of the newly developed traffic management system as well as organise data processing, cloud computing principles were used to design a distributed processing platform, referred to as ITS-Cloud in this work, which hosts all traffic management services and applications.
- **Micro-scale traffic prediction component**

  The micro scale traffic prediction is a novel component that allows the traffic management system to perform short term, highly detailed predictions of state of traffic in an intersection area. It is achieved using a purpose-designed microscopic simulation engine wrapped in a cloud service.
- **Meso-scale traffic prediction component**

  The meso-scale traffic prediction component is able to predict occurrence of vehicle waves or platoons before they enter the sensing range of an intersection.

Using such a service the intersection controller can usually adjust the signalling stages such that the vehicle group is allowed to pass without stopping resulting in significant reduction of energy expenditure.

- **Multi-dynamic service allocation**

  The specific requirements of traffic management applications have led to modification of a typical service allocation by allowing multiple users to connect to dynamically allocated service instances. The new allocation method is referred to as multi-dynamic service allocation and services created using such a method are referred to as multi-dynamic services in this work.
- **Cloud-integrated microscopic traffic simulator**

  A new microscopic traffic simulation tool has been developed throughout this project. While microscopic traffic simulation is not new, the novelty lies in integration of the simulated traffic network components with the cloud platform. Such an integration provides the ITS-Cloud services with common access methods to sensor data and presents traffic system components such as lane sensors or traffic lights as ITS-cloud services. This is based on a premise that when the developed traffic management system is deployed in the real world, lane sensors, intersection controllers and other infrastructure components will be equipped with software wrappers that would present their functionality to the cloud system in a uniform manner.

There were several research papers published by the author throughout the duration of this research project. The concept of using cloud computing to host a traffic management system and other ITS services was introduced in:

- "Cloud Computing Concept for Intelligent Transportation Systems", *In the proceedings of the* 14th *International IEEE Conference on Intelligent Transportation Systems (IEEE ITSC 2011)*, 05 Oct - 07 Oct 2011, Washington, DC, USA, Paweł Jaworski, Tim Edwards, Jonathan Moore, Keith Burnham (see

Appendix C).

The paper dealing with the traffic management system itself, including all the contained traffic control algorithms, was published as:

- "Distributed Traffic Flow Optimisation and Control for Intelligent Transportation Systems", *In the proceedings of the International Conference on Systems Engineering (ICSE 2012)*, 11 Sep - 13 Sep 2012, Coventry, UK, Paweł Jaworski, Tim Edwards, Keith Burnham, Olivier Haas (see Appendix D).

The details of the microscopic traffic simulation tool have been published in:

- "Microscopic Traffic Simulation Tool for Intelligent Transportation Systems", *In the proceedings of the* 15^{th} *International IEEE Conference on Intelligent Transportation Systems (IEEE ITSC 2012)*, 16 Sep - 19 Sep 2012, Anchorage, Alaska, USA, Paweł Jaworski, Tim Edwards, Keith Burnham, Olivier Haas (see Appendix E).

Additionally the author implemented a data base system and a fuzzy logic longitudinal controller for the Network Assisted Vehicle (NAV). NAV is semi-autonomous vehicle developed at MIRA Ltd. Its primary purpose is to automate ITS and advanced driver assistance systems (ADAS) testing. Measurements obtained from the NAV were used to obtain the parameters of the vehicle model, see Section 6.3 of Chapter 6.

It was planned to use NAV in future real world evaluations of the traffic management solutions presented in this thesis.

- "A Network Assisted Vehicle for ADAS and ITS testing", *In the proceedings of the* 14^{th} *International IEEE Conference on Intelligent Transportation Systems (IEEE ITSC 2011)*, 05 Oct - 07 Oct 2011, Washington, DC, USA, Tim Edwards, Jonathan Moore, Maria Loukadaki and Paweł Jaworski (see Appendix F).

- ”Autonomous longitudinal control for a Network Assisted Vehicle”, *In the proceedings of the* 11th *International Symposium on Advanced Vehicle Control (AVEC ’12)*, September 2012, Seoul, Korea, Tim Edwards, Paweł Jaworski and Maria Loukadaki (see Appendix G).

1.5 Outline of approach

This document starts with a description of the overall project goals, followed by identification of relevant research areas. The project background in Chapter 2 provides an overview of existing technologies and the current state of the art in relevant research fields allowing for clear formulation and justification of the research goals. Out of applicable publications the chapter focuses around a chosen few that were deemed most important or provide a useful introduction and background to the content of this work.

Chapter 3 introduces the traffic control algorithms used in this work. Their inner working is described and sensor data requirements are defined. The benchmark algorithms are defined first followed by the novel traffic management methods.

The ITS-Cloud platform is introduced in Chapter 4. It is explained how a cloud computing system works and how it was modified and implemented for ITS applications.

Chapter 5 introduces the Cloud computing Traffic Management System (CTMS) which, using the cloud computing platform described previously, manages the traffic flow throughout the traffic network. All the cloud services that form the system are described and the two levels of traffic management are introduced.

Chapter 6 describes the traffic simulation tool developed for this PhD project. An explanation is provided to why it was chosen to develop a new tool instead of using one of the already existing simulators. Details of the implementation are provided and it is explained how individual components of the traffic network are simulated.

Chapter 7 evaluates the developed traffic management methods and other components of the Cloud based Traffic Management System using the traffic simulation tool described in Chapter 6.

Chapter 8 presents the conclusions of this work, recognises the limitations of the developed apporaches, and outlines the possible future research directions.

Chapter 2

Literature background

This chapter presents a review of the relevant underpinning research. It aims to familiarise the reader with the components and technologies that were used to build the ITS-Cloud platform, the Cloud based Traffic Management System (CTMS) and the traffic simulation tool. There are two areas of research covered by this review: traffic management and simulation covered in the Sections 2.1 to 2.5, and distributed computing discussed in Section 2.6.

- The first section provides an overview of criteria and objectives used by traffic management techniques.
- The second section reviews the state of the art in traffic management focusing on traffic control techniques and algorithms that inform the new traffic management scheme developed in Chapter 3.
- The next two sections provide essential background to the realisation of the microscopic traffic simulation presented in Chapter 6.
- Section 2.3 reviews existing traffic flow models as well as selection of the most relevant traffic simulation tools exploiting those models.
- Section 2.4 reviews methods of automatic vehicle control.
- The fifth section reviews the wireless communication technologies in vehicular

environments.

- Section 2.6 reviews the cloud and grid distributed computing approaches that are used to create the ITS-Cloud distributed computing platform in Chapter 4.
- Section 2.7 outlines the identified gap in knowledge.
- The final section concludes the chapter and outlines the identified gap in knowledge.

2.1 Traffic management criteria, constraints and objectives

Before investigating traffic management methods this section provides an overview of optimisation objectives and criteria used in traffic management.

Criteria (optimisation objectives) used for traffic management have been identified as follows:

- journey time (minimise) [18, 19, 20]
- wait times/delay (minimise) [21, 22, 23]
- energy consumption and emissions (minimise) [18, 24, 25, 26, 27]
- road utilisation, throughput (maximise) [21, 25, 26, 28]
- average speed increase [29]

In order to guarantee safety and fair treatment of all traffic flow directions [22] the following constraints are imposed:

- inter-green periods (also referred to as set-up time)
- minimal green/red phase duration
- maximal red phase duration

The rules governing interaction between vehicles form an important part of traffic management. The objectives for longitudinal vehicle control have been identified as follows:

- maximise speed [30, 31]
- minimise separation gap (to increase road throughput)
- minimise variation in separation gap

The constraints for longitudinal vehicle control are as follows:

- speed limit
- minimal safe separation

2.2 Traffic management and control

Traffic management is one of the fundamental functions of the ITS and the main focus of this work. Arbitrary control over the flow of traffic is necessary to guarantee safety in situations where multiple traffic flow directions intersect. A traffic control system determines how to prioritize the flow of vehicles from different directions to clear the conflict zone. The control algorithm should guarantee a fair management and prevent unnecessary delays.

The most widely used approach involves use of traffic signals (lights) located in critical areas of the traffic network such as intersections and signalled pedestrian crossings. In the second approach, individual vehicles are influenced by a traffic controller by means or wireless communication or by variable message signs (VMS) and can react on a voluntary basis or autonomously. Such an approach is referred to as vehicle actuated traffic control (VATC) in this work.

Traffic lights can control the vehicle flow by stopping the vehicles or enabling them to pass. VATC allows finer control over the traffic enabling setting variable speed limits, advising vehicles on an optimal route and even informing the vehicles on the future state of traffic lights.

2.2.1 Traffic light based traffic control

The simplest way of governing an intersection is to use traffic lights with a pre-defined cycle plan. Such approach is often referred to as fixed timing or fixed cycles (FC) control. FC activates each traffic flow for a pre-determined period of time. The timings have be optimised off-line based on statistical data, which means that such an algorithm is unable to cope with a sudden unexpected change in traffic flow [19].

To enable a FC algorithm to react to the current traffic situation signalling stages can be dynamically increased or decreased depending on traffic demand. Such an adaptive extension to FC is utilised by the Split Cycle Offset Optimisation Technique (SCOOT) [32]. SCOOT is one of the most developed urban traffic control systems (UTC) in the world with several successful deployments in the UK and worldwide [19]. The system works by optimising the length traffic cycle stages in all intersections in a given region. Such an approach prevents the intersection controller from reacting quickly to high magnitude changes in traffic flow, however the inherent safety of an incremental system makes it robust to erroneously detected vehicles or sudden but short traffic fluctuations.

One of the greatest advantages of FC is that information about the future state of the traffic lights is available in advance at all times. Such knowledge can be exploited to optimise a vehicle approach to the intersection [10, 33].

FC traffic control is simple but usually effective for routine situations but it cannot respond appropriately to unforeseen changes in demand caused by incidents or special events. FCs rely on appropriate commissioning and maintenance to update the statistical models underpinning the timing plans. However it is typical for such systems not to be regularly maintained, making them unable to adapt to changing circumstances.

Adaptive traffic lights control (ATLC) can be used to adapt intersection control to the traffic conditions [23, 34]. ATLC adjusts the signalling stages in reaction to the traffic conditions in order to optimise specific objectives e.g. minimise delay, energy consumption, and CO_2 emissions. It adjusts dynamically and autonomously the lengths of traffic cycle stages as a response to changing road situations.

In [35] the ATLC adjusts the stage lengths of stages in predefined traffic cycles, thereby being predictable and robust. Self-organising traffic control is capable of greater adaptability and can create its own traffic cycles depending on the demand [9, 25, 27, 34, 36, 37].

In [27] the authors propose a UTC system based on agents. Agents are autonomous entities that can make their own decisions based on their desires (objective functions), usually taking into account past behaviour of other agents as well. The global solution is constructed as a result of cooperation between agents. An agent based traffic management system proposed in [38] uses different types of agents. The traffic light agents are responsible for reading traffic situation in lanes. Such information is used by the junction agents to calculate which stage should be activated. For that purpose the authors used a lane priority index P_l, calculated with respect to the type of queued vehicle and the time the traffic light was red. It is defined as follows:

$$P_l = 10 \cdot Car + 100 \cdot Bus + 1000 \cdot Emergency + \frac{(RedDuration)^2}{100} \tag{2.1}$$

Most traffic management systems work is based on the amount of queued vehicles, without taking into account the gain from allowing the vehicles to pass without stopping. A notable exception from this group is an algorithm proposed in [34]. The algorithm is based on fluid dynamics and multi particle (microscopic) simulations to control intersections. The most notable and unique feature of this approach is that it achieves a cooperative behaviour between intersections without using any types of communication between them. The system exhibits cooperative behaviour because of how the algorithm itself is structured. It prioritises moving platoons of vehicles and anticipates their arrival to the controlled intersection, based on a valid assumption that it is more optimal to keep already stopped vehicles stationary for a longer period of time in order to let a platoon of moving vehicles through the intersection without stopping them. A green wave effect is obtained because of the fact that vehicles are released from an intersection in platoons, therefore there is a good chance they will reach next intersection in this formation and the controller on the next intersection will prioritise them because of their formation. It can be noted that in that case the vehicles themselves and their configuration against each other are carriers of cooperative information between the intersections.

It was aimed to achieve such green wave effect in this work in the ITSP intersection management method (see Section 3.2 of Chapter 3) that was based on the adaptive approach of [34] described above.

2.2.2 Vehicle actuated traffic control

Vehicle actuated traffic control (VATC) is a different approach to traffic management. Instead of only relying on traffic light stages, each vehicle is given instructions, in a cooperative manner, on the trajectory they should follow to approach the intersection [39]. Additionally individual approach profiles can be created for each vehicle to optimise waiting times and fuel consumption [33].

VATC opens great perspectives for future traffic management. However it relies on a wide uptake and improved reliability and security of the necessary technology on board vehicles as well as on the whole road network. The transition from the current form of traffic management to a fully interconnected and cooperative system should therefore be done gradually with the coming of age of relevant technology. To date, vehicle actuation methods are reduced to support role for traffic light based intersection control in urban environment.

For example approaches based on communicating vehicles following instructions received from an intersection controller were demonstrated in [40] and [33] and resulted in significant reductions in fuel consumption and CO_2 emissions. Such techniques are known under many names. Authors in [33] refer to it as Eco Cooperative Adaptive Cruise Control (ECACC) and a similar technique is called Green Light Optimised Speed Advisory (GLOSA) in [10, 11]. According to [33], [11] and [10] intersection approach optimisation conflicts with adaptive traffic light control as future state of the intersection is not always determinable.

In cases where the vehicles cannot communicate with the infrastructure but communicate with themselves, decentralised techniques such as Cooperative Adaptive Cruise Control can be adopted. CACC has been shown to increase traffic throughput and significantly reduce congestion thereby alleviating bottlenecks arising from lane drop [41] or are caused by a join ramp [16].

Besides instructing the vehicles where they should stop and how fast to travel it is possible for an ITS traffic management system to tell vehicles where to go. Active traffic control might involve re-routing of traffic to keep the load on the road network balanced. A balanced traffic network is less likely to exhibit situations where one link becomes congested while others remain under-utilised. The problem is not new and its solution usually can be obtained using a weighted graph search and traverse algorithm [42], where the intersections are vertices and roads are the graph edges. The edge weight is derived from a given road link capacity and a current load.

An interesting alternative to graph algorithms has been proposed in [13]. The authors demonstrated the use of an ant colony optimisation based algorithm for optimising the vehicle flow in an urban road network. Ant colony optimisation (ACO) is a meta-heuristic method, usually applied to static routing problems like shortest path finding [43]. The authors in [13] applied this technique to a dynamic version of this problem, where the cost of traversing a graph edge is not constant and it reflects travel time rather than physical distance.

The dynamic vehicle routing mechanism implemented in this work does not find a best path for individual vehicles but searches the entire road network for the optimal routes between every pair of intersections in the managed area. The paths are calculated using the depth-first search (DFS) algorithm in combination with dynamic edge weight calculation (see Section 3.6 of Chapter 3).

2.2.3 Vehicle detection

To enable adaptive traffic control a reliable information on the current state of traffic flow is required. Such information can be obtained using various vehicle sensing techniques.

Vehicles can be detected either by sensors that are part of the infrastructure or the positioning data can be obtained directly from the vehicles that can determine their own position and communicate it to the traffic management infrastructure [7].

Induction loops [44] are usually buried in the road and are capable of detecting the presence of a vehicle above them by sensing a change in magnetic flux caused by presence of a large metallic body. Even though the basic sensor itself is only capable of reporting the presence of a vehicle (or lack of thereof) above it, coupled with analytic software it can provide valuable information about the traffic conditions and even distinguish between distinct vehicle types e.g. car, bus, truck [22].

In [45, 46] the authors demonstrated methods of estimating the speed of passing vehicles using induction loops. The method presented in [45] relied on matching the inductive waveforms caused by the vehicles and contrary to other methods was able to provide a speed estimation without the knowledge (or assumption) of vehicle length. It was shown that the method exhibits average speed measurement error of 6.7%. In this work it is assumed that the induction loops can only detect the presence of a vehicle.

More advanced sensor types capable of classifying vehicle type, providing its position (in relation to the stop lane) and speed are referred to as ITS-sensors within this work. ITS-sensors are usually based on passive cameras (operating in visual spectrum or infrared) [47] or active radar/lidar systems coupled with image recognition software [48].

ITS-Sensors are superior to induction loops in terms of greater amount of information provided and lower installation costs, however they are less robust. The performance, range and detection rates of image recognition based sensors may vary depending on the time of day or the weather [47]

Both sensor types are available in the traffic simulator presented in Chapter 6 and are used by the Cloud based Traffic Management System described in Chapter 5.

2.3 Traffic modelling and simulation

This section examines the current state of art in traffic modelling and simulation with a view to create a new traffic simulation tool (see Chapter 6) to validate the developed traffic management schemes.

Traffic modelling is utilised by traffic management systems to predict future situations [9, 27, 39] that are then used to control the traffic flow according to a specific objective (see Sections 2.1 and 2.2). The modelling techniques are also widely used to assess performance of a traffic management systems and strategies [23, 29, 49, 50, 51].

There were four levels of refinement in traffic modelling identified. Traffic can be modelled as a general flow, using equations derived from fluid dynamics, or it can be very detailed and take each separate vehicle into account. Those two approaches are called macroscopic and microscopic respectfully [52]. Mesoscopic models are placed somewhere between macro and microscopic approaches. They usually use a macroscopic approach to model traffic flow in general, supplemented with microscopic components to investigate areas of particular interest in detail [53, 54]. Nanoscopic models takes the level of detail beyond microscopic. In terms of traffic simulation it usually means simulating the vehicles down to their sub-components such as the power-train or even its own sub-components such as the engine and the gearbox [55].

The scope of the model is chosen such that it represents the investigated situation in sufficient detail. Investigating vast motorway networks will require a macroscopic model [56], analysing details of traffic flow will require a microscopic model [57]. A nanoscopic model is needed if access to internal vehicle functions and states is required [55].

The following two subsections focus on macroscopic and microscopic models, however readers should consult [58] for a comprehensive comparison of traffic modelling techniques.

2.3.1 Macroscopic traffic models and simulators

The most prominent example of a macroscopic traffic model is called TRANSYT [52] (TRAffic Network StudY Tool). TRANSYT was developed by the Transport Research Laboratory (TRL) to be used in their SCOOT traffic control system (see Subsection 2.2.1). Although it is popularly referred to as a model, it is in fact a tool chain comprising a model and an optimisation algorithm coupled together and used to adjust the signal cycle lengths in SCOOT.

MASTER [59] is a macroscopic model that uses gas-kinetic traffic equations to model the traffic flow. The traffic flow is modelled as changes in spacial vehicle density which corresponds to pressure changes in a gas medium.

The Multi-Commodity Discrete Kinematic Wave (MCDKW) model [60] categorises the vehicles into multiple commodities. The commodities are differentiated by the paths the vehicle take (sets of origin-destination points). Kinematic wave theory is then used to simulate the multi-commodity traffic.

Another approach to modelling traffic flow involves the use of coloured timed Petri nets (CP-nets) [61, 62, 63]. A Petri net is a graph describing system states and conditional transitions between them. Petri nets and timed Petri nets traditionally use discrete events to model concurrent and real time software systems and their properties. In [63] Petri nets are combined with a discrete-event modelling language Standard ML to create Coloured Petri Nets (CPN). CPN can be used to simulate systems where concurrency, communication and event synchronisation play a major role. Other macroscopic traffic simulation models include METANET [56] and NETCELL [64].

Each of these modelling approaches has its own advantages and disadvantages and is suitable for particular applications, see [58] for more details.

A significant advantage of macroscopic traffic modelling over its microscopic counterpart is that the computational cost does not increase with the amount of vehicles simulated [60].

2.3.2 Microscopic traffic modelling and simulation

A traffic network can be modelled in micro scale using a cellular automaton [65, 66]. In this approach the traffic network is divided into cells, which can be in a predefined set of states. A cellular automaton based model is discrete in both space and time. The cellular automata based traffic model family is based on a model proposed by Nagel and Schreckenberg in [67]. From the authors surnames the family are referred to as the NaSch models. Cellular automata based traffic models are simple to implement but a fine grain representation of the traffic network requires dividing the roads into many cells. The speeds of vehicle movement are quantified and dependant on the cell granularity.

INTEGRATION [52] is a microscopic model, which reflects the behaviour of individual vehicles on their lanes using a vehicle following (platooning) approach. It is also capable of modelling traffic lights with detailed signalisation plans [68].

The Traffic Software Integrated System - Corridor Simulation (TSIS-CORSIM) [58] is a set of microscopic traffic simulation tools developed by the Federal Highway Administration, USA. Its development started in 1970s making it the oldest reviewed simulation software. Despite its age the project is still active and receiving updates. CORSIM is a microscopic simulator designed to analyse freeway and urban traffic [69]. It is achieved by combining two models: FRESIM which is used to simulate traffic on motorways and NETSIM which simulates the urban environment.

Vissim is a proprietary well established and regularly updated traffic simulation platform [69, 70, 71, 72, 73] initially released in 1994. It is capable of simulating complete traffic environments including public transport vehicles, bicycles and pedestrians [70] besides general vehicle traffic [69]. It has been used for simulating motorway networks [73] as well as combined motorway and urban traffic conditions [71, 72]. One of the most interesting features of Vissim is its psycho-physical driver model for inter-vehicle gap keeping [69]. It assumes that the driver of the following vehicle cannot exactly determine the speed of the vehicle in front, which makes keeping constant separation gap impossible and makes it oscillate.

Sumo (Simulation for Urban MObility) is a mature and robust open source set of traffic simulation tools. Its development started in 2001 in the German Aerospace Center (DLR) and the first open source version was made available in 2002 [74]. It is capable of importing road network layout data in various formats, including the very popular Open Street Map format. The Open Street Map is a community driven project that relies on users contributing their maps, usually created by mapping a route with a GPS. The Open Street Map project currently has most traffic networks mapped around the world with a good level of detail [75]. Being a set of tools, not a monolithic simulator such as Vissim, Sumo is extensible thus supporting adding new features such as modelling of emissions and noise, driver behaviour [74] and intermodal transportation [76]. Sumo also found an application in the validation of microscopic vehicle models [77].

One of most interesting features of Sumo is the capability to simulate communications between the vehicles (V2V) and between vehicles and the infrastructure (V2I). Network communication simulation is and established field of research with widely available mature simulators. Sumo relies on Network Simulator 3 (ns-3) [12] to simulate the network communication. External network simulation tools can only represent a generic data channel, without taking specifics of vehicular environment into account. Sumo uses the Traffic and Network Simulation Environment (TraNS) to address that issue. TraNS extends the capabilities of ns-3 and allows simulations of vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications in vehicular environment [78].

iTETRIS is an EU funded research project aiming to create an integrated open source simulation and evaluation platform for ITS traffic management systems [12]. It is an extension to the Sumo-TraNS coupling. The iTETRIS platform features cooperative traffic management schemes making it one of the few self-contained traffic management evaluation platforms [79].

2.4 Vehicle control methods

The previous section investigated the ways of modelling traffic flow. The simulator developed in this work (described in Chapter 6) was required to be able to simulate vehicle interaction and behaviours such as cooperative platooning and optimised intersection approaches. It was therefore required to use a microscopic scale traffic simulation model, which implies simulating vehicles individually.

This section investigates methods and algorithms used to perform automatic vehicle control.

Modelling and controlling vehicle interactions involves algorithms to determine the relative speed, acceleration and separation distance of vehicles following each other as well as a vehicle changing direction, changing lane and overtaking. Algorithms for both longitudinal (often based on adaptive cruise control) [80, 81, 82, 83, 84] and lateral [85] vehicle control exist and fully autonomous vehicles are already being tested in controlled environments [30, 86].

The development of wireless communication systems (see Section 2.5) has led authors to assume their availability to improve the performance of cooperative driving schemes.

2.4.1 Longitudinal control

Longitudinal control is a well-established field, with the publications in the field appearing as early as 1989 [81], where one of the first cooperative control strategies was proposed.

It is usually expressed in the form of the vehicle following problem, where the controller aims to adjust the speed of vehicles in a platoon. The objectives are to minimise the distance to the preceding vehicle (in order to maximise road utilisation) and to keep the separation gap constant. Those two objectives contradict each other as the small inter-vehicle gaps provoke large controller actions that reduce platoon stability [87] and may potentially result in an unsafe situation or even a collision. The concept of string stability is used to assess the stability of vehicles in a platoon [88] . A platoon is considered string stable when the inter vehicle spacing errors are guaranteed not to amplify as they propagate down the platoon [87, 89]. Longitudinal control should therefore aim to provide string stability.

The time headway spacing policy is a widely used approach to the vehicle following problem [28, 90, 91, 92, 93]. It defines the desired separation between vehicles $s_p^{desired}$ as a product between the vehicle speed V and the time headway τ [94] as follows:

$$s_p^{desired} = s_0 + V\tau \tag{2.2}$$

Some implementations include a fixed minimal separation s_0 [90].

The time headway parameter can either be constant or vary in response to the road conditions or the manoeuvres the platoon is undertaking. If a variable time headway policy is adopted, choosing or formulating the variable time headway parameter becomes the key issue of the control solution. In [90] the authors considered such a strategy to control a platoon of automated heavy duty vehicles. The authors demonstrated that the inter vehicle gap fluctuations are reduced when the time headway $\tau(\Delta V_p)$ is adjusted as follows:

$$\tau = \tau_0 - c_1 \Delta V_p \tag{2.3}$$

Where τ_0 is the base time headway, c_1 is a tuning parameter and ΔV_p is the relative speed of the vehicles. The authors compared the variable and constant $\tau = 0.7s$ time headway approaches $\tau_0 = 0.1s$ and $c_1 = 0.2m/s^2$ and noted that while their choice of parameters did not guarantee string stability it made the inter vehicle gaps more stable than the base approach.

A constant time headway policy is used in the Adaptive Cruise Control (ACC) systems present in many modern vehicles [16, 94]. Like normal cruise control systems the ACC aims to follow the set speed by the driver but ACC will slow the vehicle down and engage in vehicle following if a slower moving vehicle is encountered.

Cooperative Adaptive Cruise Control (CACC) aims to improve the string stability of a platoon hence allowing smaller time headways, which in turn lead to better road utilisation and increased throughput. String instability is caused when a vehicle overreacts to an action of a preceding vehicle. Using inter vehicle communications it is possible to gain information on the intention of the preceding vehicles therefore avoid overreactions [95] or even coordinate the manoeuvre in the entire platoon [85]. CACC is usually realised by including additional goals to the ACC objective function. Instead of just keeping a desired separation gap the controller aims to minimise the speed and acceleration differences between the vehicles in the platoon. Some implementations incorporate such information from just the preceding vehicle [41, 89, 95], others take into account just the platoon leader [33, 96], or take weighted input from all the vehicles in the platoon [80, 93].

It was shown in [97] that the communication delays in the wireless network have significant impact on string stability in CACC platoons. This means that while a CACC can provide greater platoon stability than an ACC the vehicles are still required to maintain appropriate spacing defined by the time headway policy.

An alternative to the time headway based algorithms has been presented in [15], where the authors presented a fuzzy logic approach to inter vehicle gap keeping. Such an approach enabled the authors to design an incremental controller that actuated brake and accelerator pedals using multiple input variables. The controller aimed to minimise differences in speed, acceleration and like a time headway policy aimed to maintain an appropriate separation gap.

The simulation tool developed in Chapter 6 allows for simulation of vehicles using both the constant time headway spacing policy and CACC platooning.

2.4.2 Lateral control

Lateral control deals with adjusting vehicle heading. Whilst longitudinal control is one dimensional, lateral control deals with two physical dimensions, making it more challenging to implement. Automatic vehicle control is achieved by applying both longitudinal and lateral control. The most notable examples of such full vehicle control are the DARPA Grand Challenge related vehicle projects [30, 31, 86, 98, 99, 100]. The steering control algorithm used in [86] relied on a list of waypoints and aimed to head the vehicle towards the next waypoint.

In microscopic simulation environments lateral vehicle behaviour is usually modelled by a lane changing mechanism [101]. Lane changing is usually modelled as a multi stage process where a vehicle has to first indicate the desire to change lanes. Such desire is usually triggered by encountering an obstacle, such as a slower moving vehicle, on the current lane. The vehicle completes the lane change manoeuvre after it has been established that it is safe to do so [102].

A lane changing mechanism based on the above is implemented in the traffic simulation tool created in this work (see Subsection 6.3.4 of Chapter 6).

2.5 Communication methods in the vehicular environment

Many of the techniques described in the previous sections require information to be delivered between the vehicles or from the traffic infrastructure. Techniques such as the CACC (see Subsection 2.4.1), collision avoidance [103], incident response [104], and route guidance (see Subsection 2.2.2) rely on a wireless vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication capabilities. This section provides an overview of such wireless communication solutions in the vehicular environment.

The vehicular environment is very challenging for wireless information exchange. The network consists of both mobile (vehicles) and stationary (infrastructure) communication nodes and its topology is in a state of constant change due to vehicle mobility. The environment varies from open rural areas to densely populated urban centres resulting in different signal propagation conditions. Network node density also varies in time and space, peaking in rush hours in urban environments and is low in sparsely populated areas. Such unique, in the area of wireless communication, circumstances have led to the development of several technologies to address the issues associated with wireless communication in vehicular environment.

2.5.1 Networking in vehicular environment

The Vehicular Ad-Hoc Networks (VANETs) are a loose term defining requirements for Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communications and providing list of applications [105, 106, 107]. VANETs are a specific subclass of Mobile Ad-Hoc Networks (MANETs) [108], which are meant to enable communication in situations where infrastructure is unavailable or inoperable, such as disaster zones or remote areas [109]. VANETs also consider additional services and applications that, while not directly related to safety and traffic performance, also contribute towards the driving satisfaction. Such applications are usually referred to as infotainment (information and entertainment) [110] include parking space availability advice, traffic congestion advice and weather forecasts [106].

The implementation of these networks relies on the physical communication layers to carry wireless information. Non-time sensitive communications such as congestion and route advice or parking space availability information are normally realised using mobile broadband technologies due to a large amount of information potentially transmitted [111].

Safety and traffic network performance communication are normally short but frequently repeated messages such as CACC messages, collision warnings, and information on the traffic light changes. The safety critical communication can neither rely on the availability of the infrastructure nor tolerate unpredictable delays associated with public data networks. The IEEE 802.11p standard, also referred to as Wireless Access in Vehicular Environments (WAVE), defines the physical layer of communication in vehicular environment [112]. It operates on a frequency spectrum band called Dedicated Short Range Communications (DSRC) allocated exclusively for V2V and V2I messaging. The allocated spectrum is located in 5.9GHz frequency range and is 70Mhz wide [113] and is divided into seven 10MHz wide channels, which corresponds to channels 172 to 184 using the IEEE 802.11 channel numbering scheme. There are four service channels carrying the general V2x traffic, a control channel is reserved for safety communication only. A high power public safety channel is usually used by the roadside equipment to broadcast road state and safety information. A critical 'safety of life' channel is meant to be used in emergencies only. The use of the remaining channels remains to be defined.

2.6 Distributed processing and its applications

The previous sections discussed various traffic management and vehicle control techniques with the aim to build a knowledge base to design a robust and effective adaptive traffic management system (ATMS). Every ATMS is tasked with data collection from geographically distributed sensors and controlling traffic lights at different intersections. The concepts of advanced ATMS assume that wireless communication nodes (V2I) are used to send commands to vehicles. Such nodes also have to be geographically distributed in order to provide adequate coverage. Each sensor, intersection controller and V2I node has to be equipped with some processing capability to perform its function. This means that ATMSs are distributed processing systems and distributed computing principles should apply to them. This section aims to provide background on distributed processing systems and their applications. The possibility of applying such techniques in the field of traffic management is investigated.

A distributed processing systems evolved from parallel computing. Historically parallel computing was introduced to help solve complex mathematical problems that otherwise would not be solvable on a single machine in an adequate amount of time [114].

In such systems the computational resources were assembled into large groups of statically linked processing nodes called clusters, which possessed a large amount of processing power, but were not very scalable. As the processing power and the amount of users of parallel computing grew a need for more scalability became evident. A grid computing system was proposed in [115].

Grid computing aims to provide a scalable distributed computing environment meant for solving large, computationally intensive problems. One of the most prominent users of grid computing technology is the European Center for Nuclear Research (CERN), which uses the extensive computation capabilities of a grid system to process data gathered from various experiments [116].

Cloud computing, although very similar to grid computing in many aspects, aims to provide a universal execution platform for various applications with the primary focus being the scalability and transparency of the platform.

Both cloud and grid systems are service oriented architectures (SoA) [117]. The grids follow the software as a service (SaaS) approach. The cloud makes it possible to expose services at three different levels: SaaS, platform as a service (PaaS) and infrastructure as a service (IaaS) [118]. SaaS provides users with access to purpose built software components. PaaS provisions access to a high level platform that can be used by the users to deploy their own services. IaaS provides users with access to hardware platforms with managed (often priced) access to resources enabling them to deploy wide variety of software.

Table 2.1 provides comparison of selected cloud and grid system features [119]. The cloud has been considered an extension or a superset of the grid [120] or its user-friendly version [121]. The services, although usually being complex applications,

Table 2.1: Comparison of cloud and grid systems

Feature	**Grid**	**Cloud**
Resource sharing	Shared resources	Assigned resources are not shared
Resource heterogeneity	Aggregation of heterogeneous resources	Aggregation of heterogeneous resources
Virtualisation technology	Data and computing resources virtualised	Hardware and software platforms virtualised
Security	Through credentials delegation	Through isolation
Architecture	Service oriented (SoA)	SaaS, PaaS or IaaS
Centralisation degree	Decentralised control	Centralised control
Usability	Hard to manage	User friendly
User access	Transparent access for end user	Transparent access for end user
Software workflow	Pre-defined workflow required	Automatic workflow for most applications

are considered basic components of the cloud or grid system and their invocation is considered an atomic (undividable) operation. Both architectures differ in the way services are used and allocated. In many grid systems, a service is usually a direct access to computational power in the form of execution of a parallel client application on each node [118], which is just one step ahead from using a standard cluster computer. More advanced grid systems offer dedicated services that can be combined to create grid applications similarly to the standard applications which are composed of a collection of processor instructions.

Both cloud and grid computing approaches have been adopted in this work to create the ITS-Cloud distributed processing platform. The ITS-Cloud, see Chapter 4, was designed to host the Cloud based Traffic Management System (CTMS) and other ITS applications. Such purpose oriented design required adaptation of most appropriate approaches from both cloud and grid computing models.

The ITS-Cloud adopts the general distributed processing concepts common to both cloud and grid systems including flexible messaging system as well as service discovery, access and lifecycle management.

The grid computing specific features used within the ITS-Cloud include:

- Static services which can be shared between multiple users. In the ITS-Cloud such serives represent prysical hardware components and are always present in the system.

Features related to cloud computing include:

- Dynamic services which are allocated on demand for exclusive single user use.
- The novel multi-dynamic service allocation method providing the benefits of dynamic allocation, duplication and portability as well as making the service acessible to multiple users, see Chapter 4 for a detailed discussion.
- Service containers that magage the life cycle and connectivity of the dynamic and multi-dynamic services.

2.7 Discussion

ATLC and VATC have been stated as incompatible with each other [10, 33]. Existing vehicle control schemes operate by applying the benefits of knowing the changes of traffic cycle in advance to optimise behaviour of individual vehicles as a response to the future signal phase change [24, 122]. ATLC endeavours to optimise the traffic cycle durations, often offering green light extensions to when more approaching vehicles is detected [8, 9, 34, 35, 57, 65]. Both those approaches assume that the behaviour of the affected party is easily predictable. However, increasing complexity of ATLC and VATC make their behaviour becomes more difficult to predict. The challenge is for ATLC and VATC to cooperate. A behaviour to avoid would be to have a smart vehicle approaching an intersection deciding to decelerate owing to predicted end of the green light stage, whilst ATLC adapting stages to grant the given vehicle green stage. This situation would result in a suboptimal solution resulting in unnecessary vehicle deceleration and hence increased journey time and energy consumption. This identified issue caused by the ATLC traffic stage switch decisions being implemented immediately [34] without considering the adjustments made by the vehicle prior to the ATLC changing its behaviour, thereby preventing the vehicles to optimise their behaviour in relation to traffic signalling cycle [24].

The main goal of this research is to bridge this gap and enable simultaneous use of ATLC and VATC.

2.8 Conclusion

This chapter provided an overview of background knowledge on various approaches, methods and technologies required for understanding of ITS traffic management. Means of exercising control over intersections and vehicles have been examined identifying a wide range of different approaches. A compatibility issue between the traffic light based control and vehicle actuated traffic control has been identified to occur when an adaptive intersection control scheme is used. Solving that incompatibility issue is the main goal of this work.

Methods of simulating traffic flow their scopes and applications were examined leading to a choice of the microscopic modelling for development of a new traffic simulation tool. Such choice prompted examination of vehicle simulation and control methods.

Wide use of wireless communication techniques in most ITS applications led to a survey of wireless communication techniques in vehicular environment being carried out.

Finally the cloud and grid distributed computing approaches were described and compared paving a way towards implementation of the ITS-Cloud platform.

Chapter 3

Traffic Management Methods

The previous chapter provided the background to this project. Existing methods of improving traffic flow have been identified and discussed. A gap in knowledge, which prohibits the simultaneous use of adaptive traffic light control and vehicles intersection approach optimisation, has been identified.

This chapter addresses this gap and proposes the new Two-Step traffic optimisation method as well as other improvements to existing algorithms. The chapter is organised as follows:

- The first section describes the fixed cycles and induction loop based queue minimisation techniques. Those traffic management methods form the benchmark that is used to evaluate the novel traffic management method in Chapter 7.
- The second section introduces a modified pressure based intersection management algorithm (ITSP) that is used as the stepping stone to the Two-Step traffic optimisation method.
- The third section describes the main novelty of this work: the Two-Step traffic management algorithm. The requirements and enabling components for such a method are discussed.
- The fourth section describes the Intersection Approach Trajectory Optimisation

(IATO), a vehicle based traffic control mechanism enabling advising vehicles of what speed they should aim to travel at.

- The fifth section provides a short case study for the Two-Step and IATO mechanisms.
- The sixth section describes the area management approach aiming to avoid congestion by suggesting alternative routes to the vehicles.
- The last section concludes the chapter.

3.1 Benchmark algorithms

This section describes the benchmark intersection control algorithms (ICAs) used in this work. Their purpose is to provide a reliable point of reference when investigating the performance of the novel Two-Step traffic management method or the combined traffic management methods in the CTMS.

3.1.1 Algorithm 1: Fixed cycles intersection control algorithm

Each traffic cycle comprises several signalling stages and each stage is composed of one or more non-conflicting traffic flow directions.

In the fixed cycle based control the cycle and its stages are of fixed and predefined duration. Each cycle plan CP is composed of at least two signalling stages g_i of defined duration $t_{g,i}$. The signalling cycle can be written as:

$$CP = \{\{g_1, t_{g,1}\}, \{g_2, t_{g,2}\}, ..., \{g_n, t_{g,n}\}\} \tag{3.1}$$

Where each stage consists of compatible flows:

$$s_i = \{f_{i|1}, f_{i|2}, ...\} \tag{3.2}$$

The process of generating signalling stages is discussed in Section 6.5 of Chapter 6. Each flow $f_{i|j}$ represents a single traffic flow direction governed by a single set of traffic lights.

The intersection management component converts the absolute stage durations $t_{g,i}$ into stage starting times $T_{g,i}$ relative to the beginning of the cycle as follows:

$$\begin{aligned} T_{g,1} =& \quad 0 \\ T_{g,i+1} =& \quad T_{g,i} + t_{g,i} \end{aligned} \tag{3.3}$$

The signalling cycle and its stages are illustrated in Figure 3.1.

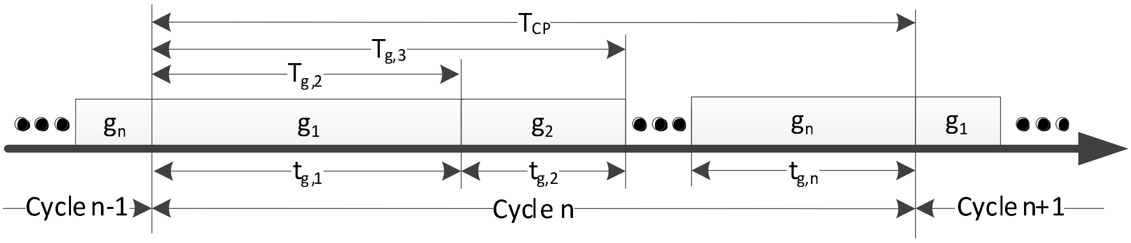


Figure 3.1: A repeating traffic light cycle divided into fixed length stages.

The total cycle length T_{CP} is the sum of all stage durations:

$$T_{CP} = \sum_{i=1}^{n} t_{g,i} \tag{3.4}$$

In order to determine which stage should be active at any time the current cycle time is compared with relative stage starting times (see Algorithm 1).

```
Procedure FC():
$cycleTime = currentTime \textbf{ mod } T_{CP}$
for $i$ = 1 to $n$ do
    if $cycleTime \geq T_{g,i}$ and $cycleTime < T_{g,i+1}$ then
        flag stage $i$ as active
    end
end
```

Algorithm 1: Fixed cycles intersection control algorithm

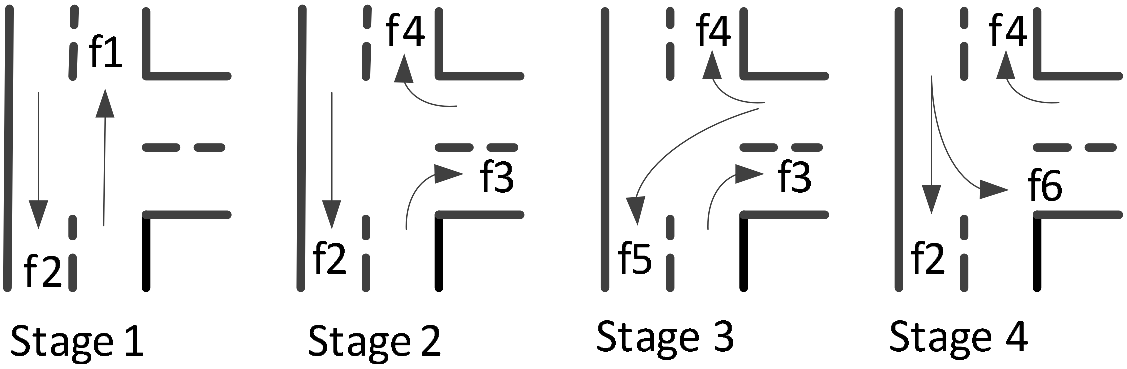


Figure 3.2: An example of intersection configuration. A cycle comprising of four stages.

Figure 3.2 shows an example of an intersection configuration. This T-shaped junction is has 6 possible flows $f_1, ..., f_6$ organised into four stages:

$$\begin{aligned} g_1 &= \{\{f_1, t_1\}, \{f_2, t_2\}\} \\ g_2 &= \{\{f_2, t_2\}, \{f_3, t_3\}, \{f_4, t_4\}\} \\ g_3 &= \{\{f_3, t_3\}, \{f_4, t_4\}, \{f_5, t_5\}\} \\ g_4 &= \{\{f_2, t_2\}, \{f_4, t_4\}, \{f_6, t_6\}\} \end{aligned} \tag{3.5}$$

3.1.2 Algorithm 2: ILC intersection control algorithm

The 'Induction Loop Counter' (ILC) is an adaptive intersection management algorithm. It requires each lane leading to the intersection to be equipped with an induction loop, preferably in an appropriate distance from the stop line. It is assumed that each induction loop is equipped with a counter that counts passing vehicles. If the signal associated with the sensor shows a red light it is assumed that all the vehicles that passed the induction loop became queued.

It was shown in [19, 123] that in congested traffic conditions the highest traffic performance is achieved by reducing the amount of switch overs as much as possible. Following this principle the algorithm uses a queue discharge model that anticipates the time needed to clear it. The time to clear a vehicle queue on an inbound link is denoted as T_j^Q where the subscript j corresponds to the traffic stage number in the given intersection.

The queue discharge model is a two dimensional lookup table that when indexed by the queue length denoted q_j and the speed limit denoted $V_{max,j}$ provides an estimate of the queue discharge time and is denoted as $M(q_j, V_j)$. The lookup table was populated experimentally using a traffic simulator by measuring the times it takes on average to discharge vehicle queues of different lengths under different speed limits. Therefore the queue clear time for induction loop j can be written as:

$$T_j^Q = M(q_j, V_j) \tag{3.6}$$

The estimation of queue clear time is performed separately for each induction loop sensor (therefore for each inflow lane) and the stage associated with the lane with the longest queue, denoted T_{max}^Q, is activated for the amount of time it was predicted to be needed to clear it increased by the time it takes to switch stages T_{switch}.

$$T^Q_{max} = max_j(T^Q_j) + T_{switch} \tag{3.7}$$

Once the active stage expires T^Q after its activation the algorithm is run again. Depending on the current traffic situation it can either make a decision to prolong the activity period of a current cycle or switch to a different one (see Algorithm 2). Parameter n represents the amount of stages in the cycle plan CP and m is the amount of individual queues (on traffic flows measured by the induction loops) in each stage. After the procedure is finished the stage number indicated by the *stageNumber* variable is activated.

```
Procedure ILC():
stageNumber = 0
maxQueueClearTime = 0
for i = 1 to n do
    maxStageQueue = 0
    for j = 1 to m do
        if maxStageQueue < $T^Q_j$ then
            maxStageQueue = $T^Q_j$
        end
    end
    if maxStageQueue > maxQueueClearTime then
        maxQueueClearTime = maxStageQueue
        stageNumber = i
    end
end
Activate signalling stage stageNumber
```

Algorithm 2: ILC intersection control algorithm

3.2 ITS Pressure intersection control algorithm

The idea of using pressure indexes to reflect the situation on roads was described in [34]. The authors defined the pressure priority indexes π_i for each signalling stage i as follows:

$$\pi_i = \frac{\hat{n}_i}{\tau_{i,\sigma}^{pen} + \sigma_i + \hat{g}_i} \tag{3.8}$$

Where $\hat{n}_i$ is the amount of vehicles expected to pass the intersection during the green time period $\hat{g}_i$. Parameter $\tau_{i,\sigma}^{pen}$ represents the penalty for terminating the currently active signalling stage and σ_i is the switch-over time.

The 'ITS Pressure' (ITSP) modifies this approach to incorporate the knowledge of speeds and locations of the approaching vehicles. Once the necessary vehicle data has been obtained from the ITS sensors (see Subsection 2.2.3 of Chapter 2) the algorithm calculates the pressure index $\pi_{i,j}$ exercised on the stop line by each vehicle i on inflow lane j.

$$\pi_{i,j} = \frac{d_{v,T}}{max(0, d_{i,j} - d_{v,T}) + d_{v,T}} (1 + \beta_V V_{i,j}) \tag{3.9}$$

The pressure associated with a vehicle increases with its speed $V_{i,j}$ and with the decreasing distance to the stop line $d_{i,j}$. Parameter β_V adjusts the weighting of the speed (value of 0.1 was used in this work). The maximal distance related pressure is achieved when the vehicle is within a critical distance $d_{v,T}$ from the intersection. This is meant to emphasise the fact that once a vehicle passes a certain point it will be forced to slow down due to the time it takes to switch signalling stages. The critical distance is given as follows:

$$d_{v,T} = T_{switch} V_{max,j} \tag{3.10}$$

Where $V_{max,j}$ is the speed limit on lane j and T_{switch} is the traffic light switching time.

The signalling stages are composed of inflow lanes as defined in Equation (3.2) in Subsection 3.1.1. It is worth remembering that each lane can belong to multiple signalling stages.

The total pressure index Π_s for signalling stage s is given as:

$$\Pi_s = \sum_{j \in s} \sum_{i \in j} (\pi_{i,j}) \tag{3.11}$$

A new traffic stage S_{new} will be activated if its pressure is greater than the pressure on the current stage $S_{current}$. A switching threshold H^{π}_{switch} is used to account for the cost of switching (see Algorithm 3).

During stage switch-over no vehicles can enter the intersection, which results in wasted road capacity. The amount of wasted capacity depends on the speed limit, the higher the speed limit the more vehicles could have passed the intersection. In order to account for that, as well as to limit the amount of stage switch overs, a minimal stage duration $T_{s,min}$ is defined as follows:

$$T_{s,min} = \alpha_{V,1} V_{max,j} + \alpha_{V,0} \tag{3.12}$$

The parameters $\alpha_{V,0} = -4000$ and $\alpha_{V,1} = 600$ were used in this work, which result in minimal stage times of 5 seconds for the speed limit of 15m/s (smallest investigated speed limit) to 14 seconds for speed limit of 30m/s (highest investigated speed limit).

The algorithm was designed to prioritise moving vehicles with aim to conserve energy by allowing them to pass without stopping. This can however lead to situations where already stopped vehicles will never be allowed to pass if there is a constant stream of moving vehicles arriving from other directions. To avoid such situations the intersection controllers are equipped with a supervisory mechanism that ensures that each direction will be given green light with minimal defined frequency.

A green wave effect is usually obtained either by synchronizing gap timings between adjacent intersections or by using a centralized controller governing several intersections [23]. Even though there is no explicit green wave negotiation between intersections governed by ITSP, the algorithm was designed to exhibit green wave effect automatically. It is due to the fact the upstream intersections usually release vehicles in platoons or in large groups. Such a group of moving vehicles is interpreted as an area of high ITS-pressure by the algorithm and usually results in a traffic stage switch in favour of the approaching vehicle group.

The most significant disadvantage of this algorithm is the fact that the traffic

```
Procedure ITSP():
for each stage $s$ do
    stagePressures[s] = 0
    maxPressure = 0
    maxPressureStage = 0
    for each inflow lane $j$ in stage $s$ do
        lanePressures[j] = 0
        for each vehicle $i$ on lane $j$ do
            $\pi_{i,j} = \frac{d_{v,T}}{max(0,d_{i,j}-d_{v,T})+d_{v,T}}(1+\beta_V V_{i,j})$
            lanePressures[j] = lanePressures[j] + $\pi_{i,j}$
        end
        stagePressures[s] = stagePressures[s] + lanePressures[j]
        if $stagePressures[s] > maxPressure$ then
            maxPressure = stagePressures[s]
            maxPressureStage = s
        end
    end
    if $maxPressureStage \neq currentStage$ then
        if $maxPressure > stagePressure[currentStage] + H^{\pi}_{switch}$ then
            switch signalling stage from currentStage to maxPressureStage
        end
    end
end
```

Algorithm 3: ITSP intersection control algorithm

management decision it generates is optimal only at the moment it is made, therefore it has to be implemented immediately, leaving no time to advise the vehicles on the upcoming signal change. The Two-Step traffic optimisation technique described below aims to address this issue.

3.3 The Two-Step Traffic Optimisation Method

In order to enable the intersection controller to advise incoming vehicles whilst at the same time responding to varying traffic timely it is proposed to use a Two-Step process as follows:

1. Generate intersection management decision:
    - Perform a short term prediction of the traffic situation (H_p time units ahead).
    - Use the predicted situation image as an input to the ITSP intersection management algorithm in order to obtain intersection management decision.
    - Delay implementation of the obtained intersection management decision by H_p.
2. Having established the future state of a traffic light at step 1 optimise the approach trajectories of the vehicles.

The prediction horizon, denoted H_p, determines how far into the future should the prediction be made. Longer prediction horizons lead to more time being available for optimising the vehicle approach to the intersection, however the predictions tend to loose accuracy with large prediction horizons which negatively affects traffic performance.

In Two-Step ICA the prediction horizon is selected automatically based on the intersection configuration and road conditions. It is obtained by choosing a minimum from partial prediction horizons calculated for each lane leading to the given intersection as follows:

$$H_p = min\left[\delta_H \frac{d_s}{V_l}\right] \tag{3.13}$$

In order for the algorithm to provide timely reaction the prediction horizon cannot be greater than the time between a vehicle entering the sensing range to it reaching the stop line by travelling at the maximum allowed speed. This time is obtained for each lane by dividing the sensing range d_s by the speed limit V_l, see Equation (3.13). The parameter δ_H was meant to adjust the prediction horizon depending on traffic intensity. In the current version of CTMS it has been empirically set to 0.8 and the automatic selection of δ_H is considered in future work.

The second step of the Two-Step traffic optimisation process is to generate an intersection management decision using the predicted situation image. This is done using the ITSP ICA described in the previous section, however instead of using the current situation image, the predicted image is used as the input to the algorithm. The prediction is obtained using a CTMS component called the Micro Scale Prediction Service (MiSPS, see Section 5.5 of Chapter 5).

The advice horizon H_a can therefore defined as the amount of time that is available before the generated traffic management decision is due to be implemented. The advice horizon is obtained subtracting the processing time T_p from the prediction horizon H_p:

$$H_a = H_p - T_p \tag{3.14}$$

The processing time T_p includes the communication time and is measured every time the Two-Step algorithm is executed and can vary due to external factors such as the load in ITS-Cloud and conditions in the base data exchange network.

Implementation of the calculated intersection management decision is delayed by the advice horizon allowing the vehicles to receive advice of the intersection controller's intent so their approach to the intersection can be optimised, a process that is managed in CTMS by the Intersection Approach Trajectory Optimisation (IATO). The Two-Step method is summarised in Algorithm 4.

```
Procedure Two-Step():
Note processing start time:
    $processingTimeStart = currentTime$
Calculate prediction horizon:
    $H_p = min\left[\delta_H \frac{d_s}{V_l}\right]$
Predict traffic situation in $H_p$:
    $\{\hat{\pi}_{1,1}, ..., \hat{\pi}_{i,j}\} = \text{MiSPS}(H_p, \{\pi_{1,1}, ..., \pi_{i,j}\})$
Use ITSP to determine next signalling stage:
    $nextStage = \text{ITSP}(\{\hat{\pi}_{1,1}, ..., \hat{\pi}_{i,j}\})$
Measure processing time and determine advice horizon:
    $T_p = currentTime$ - $processingTimeStart$
    $H_a = H_p - T_p$
Start IATO advice process:
    IATO($nextStage$, $currentStage$, $H_a$)
Delay implementation of stage change by $H_a$:
    wait for $H_a$
    switch signalling stage from $currentStage$ to $nextStage$
```

Algorithm 4: Two-Step traffic management method

3.4 Intersection Approach Trajectory Optimisation

This section introduces the Intersection Approach Trajectory Optimisation (IATO), which is a traffic management system component that aims to optimise how the vehicles approach intersections.

Most adaptive traffic management systems optimise traffic flow by setting the traffic lights in an appropriate manner. It has been noticed that additional traffic optimisation can also be achieved by advising the vehicles on the road situation before they enter the affected area. Such information can either come from the infrastructure or other vehicles that are already in the affected area or are participating in the incident themselves [124]. The infrastructure can provide information on current or future status of the road, such as the state of the road surface or variable speed limit.

The Intersection Approach Trajectory Optimisation (IATO) is a CTMS component that aims to provide vehicles with advance information about the intersection controller's intent. IATO relies on the Intersection Control Algorithm (ICA) to provide advance traffic light timing information. Before introduction of the Two-Step method only FC was able to provide such information.

In order to perform its function IATO requires to obtain the following information form the intersection controller:

- Current traffic stage
- Next traffic stage
- Current stage remaining time

Using that information IATO is able to determine stages that are affected by the switch and issue appropriate advices the vehicles on the affected roads or lanes. Regardless of the amount of stages an intersection has, only two of them will be involved in a switch and will receive detailed IATO instructions. Stages not involved in a switch will not have their next cycle defined and will continue to follow their default behaviour. On the stages involved in the switch-over the vehicles will be advised on the intersection controller's intent, as illustrated in Figure 3.3.

Owing to the fact that the optimal approach trajectory may be different for different vehicle types the final decision on how to approach the intersection has to be calculated on board of each vehicle. IATO provides the vehicles with the time remaining before the switch, the direction of switch (green to red or red to green) and the intersection's location. It is assumed that ITS-equipped vehicles

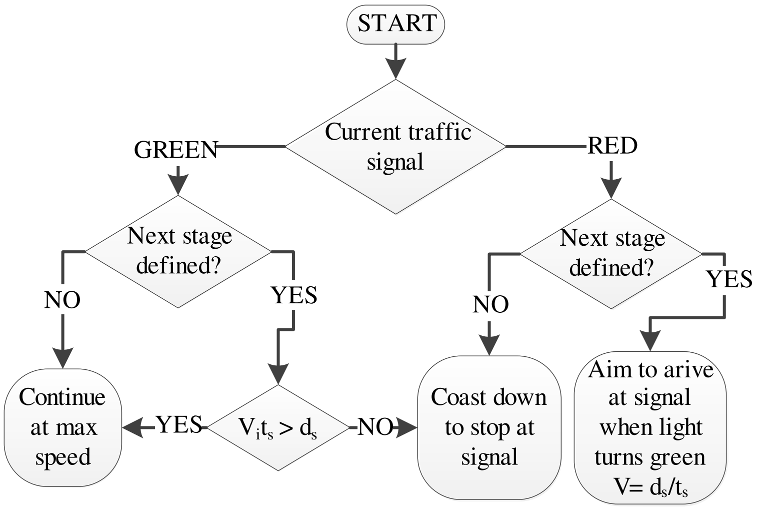

START
Current traffic signal
GREEN
RED
Next stage defined?
NO
YES
Next stage defined?
NO
YES
Continue at max speed
YES
V_i t_s > d_s
NO
Coast down to stop at signal
Aim to arive at signal when light turns green V= d_s/t_s

Figure 3.3: A diagram describing the decision process of the IATO mechanism.

are able to determine their own position, therefore distance to the intersection (d_s) approximated by a straight line can be obtained. If the vehicle is advised that the light is about to turn green, it's objective is to conserve as much kinetic energy as possible and arrive at the stop line the moment the light turns green.

In case where the light is about to turn red the vehicle has to estimate if it is going to arrive at the stop line before the traffic light switching occurs. The vehicle should continue at current or if possible at greater speed if the distance the vehicle can travel before the signal changes is greater than the distance to the stop line:

$$Vt_s > d_s \tag{3.15}$$

V denotes the current vehicle's speed, t_s is the time before signal change and d_s is the distance to the signal (stop line).

If it is determined that the vehicle will not be able to cross the intersection before the signal change it will have to stop at the stop line. Knowing where the vehicle is supposed to stop allows to optimise its deceleration profile. Optimal deceleration profiles differ between vehicle types.

Standard vehicles are unable to recover kinetic energy through braking therefore their optimal deceleration profile is dictated by the air drag. Such vehicles will coast down towards the intersection and use their brakes to stop if necessary or crawl to the intersection in case their speed is reduced to $3\frac{m}{s}$ by the friction forces before they arrive at the stop line.

In case of hybrid electric vehicles (HEV) it is more complicated. Deceleration profiles for HEVs are affected not only by the dynamics of vehicle but also by state of charge of the on board battery [125]. Therefore a HEV following its own optimal deceleration profile can force its follower to follow a suboptimal deceleration profile. A mechanism for obtaining a deceleration profile for an entire platoon is considered for future extension of this work.

3.5 The Two-Step process illustration

This section illustrates the Two-Step traffic management approach using a simple intersection with two signalling stages.

The pressure indexes, obtained using Equation (3.9), are denoted as π_1 for stage 1 and π_2 for stage 2 (see Figure 3.4). At time t_a a prediction is made H_p time units into the future and it is decided to switch from stage 1 to stage 2. The IATO is used to advise the vehicles of the imminent stage change. The vehicles on stage 1 that will not manage to pass the intersection decelerate reducing the pressure. The opposite occurs on the road belonging to stage 2. The observed pressure will be higher due to the vehicles arriving at the stop line with greater speed, as dictated by the IATO (see Section 3.4).

After a decision to switch stages is made the system is committed. Similarly to ITSP the Two-Step method is subject to minimal stage lengths $T_{s,min}$ as given by Equation (3.12) in Section 3.2. The next prediction is made at time $t_b = t_a + H_p +$

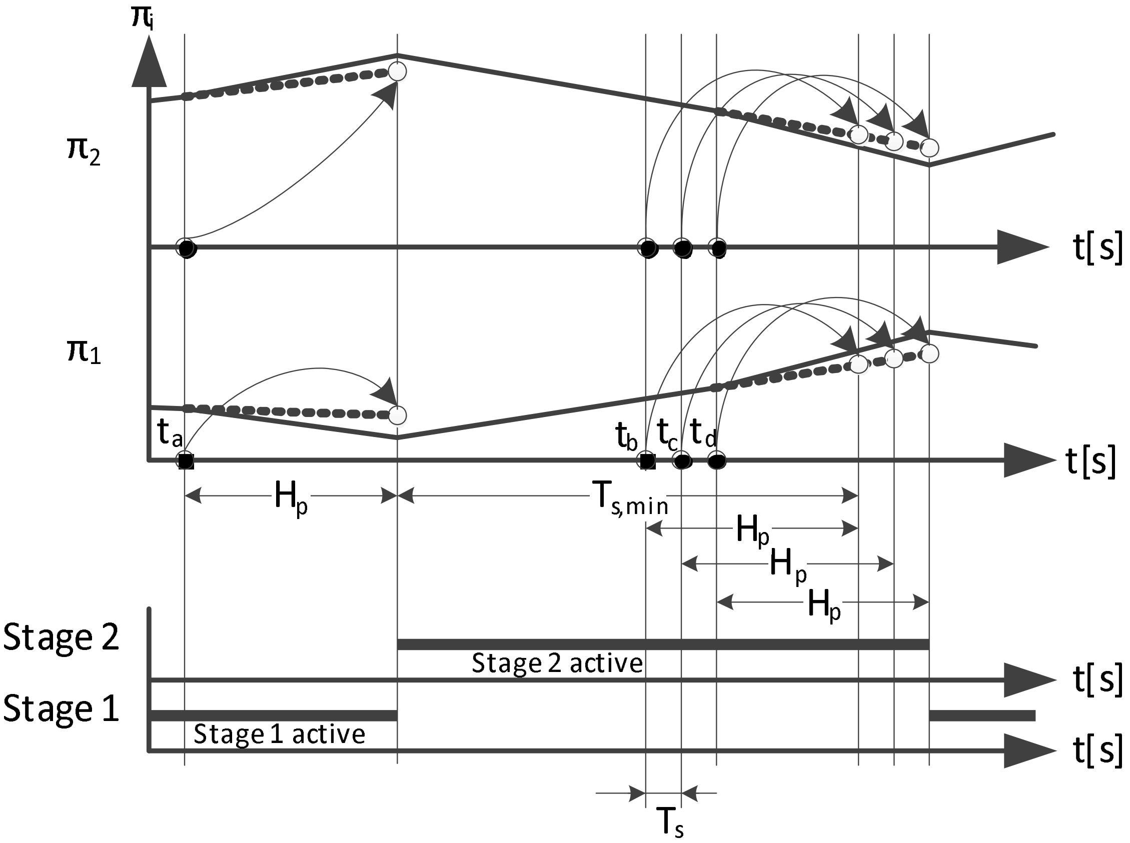


Figure 3.4: Two-Step traffic management diagram illustrated using hypothetical road situation. The first stage switch decision to switch stages, form stage 1 to stage 2, is made at t_a *and implemented at* $t_a + H_p$*. The decision to switch stages again made at time* t_d *and the change occurs at* $t_d + H_p$*.*

$T_{s,min} - H_p = t_a + H_p$. In the example it is decided not to switch stages at this point as π_2 is still greater than π_1. Due to the fact that at this moment of time the minimal green time has already passed the controller is free to change stages at any point. Therefore it will monitor the situation and changes stages when appropriate. The controller decides to switch from stage 2 to stage 1 at t_d and the switch is implemented at $t_d + H_p$. This example assumes that processing time is negligible (see Section 3.3) therefore the advice horizon H_a is equal to H_p, see Equation (3.14).

3.6 Dynamic routing

The previous section described the IATO, a mechanism used to advise vehicles on how to optimise their approach to intersections.

This section introduces another vehicle actuation based traffic management technique. The dynamic routing (DR) component monitors the state of the traffic in the road network and is able to provide route advice to appropriately equipped vehicles.

The DR mechanism maintains a weighted directed graph representation of the road network. Each road is represented by up to two edges, one in each direction. Each intersection is represented by a sub-graph, where the vertexes represent links to roads in and out of the intersection, and edges define the possible intersection traversal ways. The vertices, where the internal intersection representation comes in contact with the edges representing the links between intersections, are referred to as node-road bindings (NRB) in this work. The NRB vertexes that are connected to inflow roads are referred to as inflow-NRB (iNRB) and similarly those that are linked with intersection outflow roads are referred to as outflow-NRB (oNRB). Such representation accounts for the different intersection configurations e.g. not every turn direction is present. A graph representation of the traffic network used by DR is illustrated in Figure 3.5

The edges that define turns inside the intersection are not weighted (have zero weight), and those between the intersections are weighted to represent the traversal cost of such link. Those traversal costs have to be kept up to date to reflect the current traffic conditions on each link and are calculated as follows:

$$W_j = L_j \Upsilon_L + (VC)_{j|t} \Upsilon_{(VC)} + (VF)_{j|t} \Upsilon_{(VF)} \tag{3.16}$$

Where W_j is the cost of traversing the link subscripted j, L_j is the length of the link in meters, $(VC)_{j|t}$ is the mean vehicle count on the link calculated using last t time

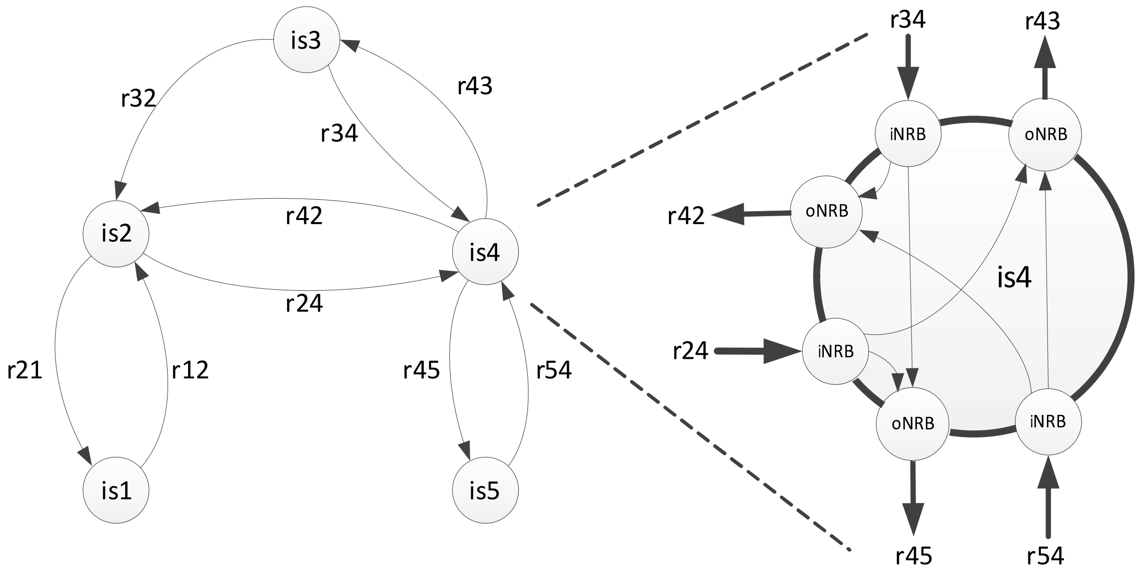


Figure 3.5: A graph representation of a road network with five intersections.

units of traffic data. $(VF)_{j|t}$ is the mean vehicle flow, given in vehicles per minute, calculated over the same time period as vehicle count. Parameters Υ_L, $\Upsilon_{(VC)}$ and $\Upsilon_{(VF)}$ are experimentally chosen weights used tune the DR mechanism.

Contrary to existing approaches [13, 126, 127] this DR mechanism does not provide vehicles with a complete route plans. Instead the vehicles are advised which turn to take as they approach each intersection. Such an approach allows adjusting the routes as traffic conditions change and takes away the burden of path following from the in-vehicle unit, however it is applicable only in urban environments where every intersection is appropriately equipped to provide such a service.

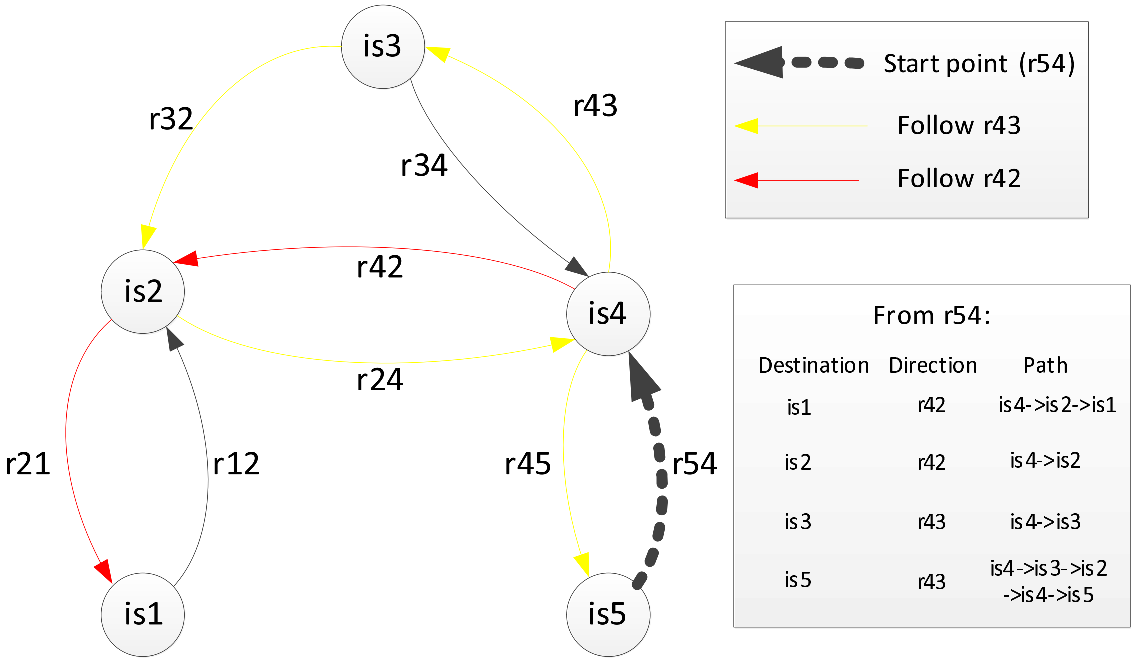
In order to perform its function the DR system maintains a list of associations between possible destinations and which oNRB should be used to reach them using optimised route. Such a list is referred to as *DR-List* in this work. A separate *DR-List* is created and maintained individually for each iNRB. *DR-List* objects are generated using a depth-first graph search (DFS) algorithm modified to work with such data representation. The ITS-DFS algorithm was implemented in a recursive manner, see Algorithm 5.

```
Procedure ITS-DFS(Current-node, Current-cost, Direction, DR-List):
for each edge from Current-node do
    Current-cost = Current-cost + cost of traversing the edge
    edge-destination = the node edge leads to
    if mapping for edge-destination exists in DR-List then
        if Current-cost is smaller than existing cost then
            replace the existing mapping of edge-destination with:
            edge-destination maps to Direction, Current-cost
            Current-node = edge-destination
            recursively call ITS-DFS(Current-node, Current-cost, Direction,
            DR-List)
        else
            return
        end
    else
        create new mapping for edge-destination:
        edge-destination maps to Direction, Current-cost in DR-List
        Current-node = edge-destination
        recursively call ITS-DFS(Current-node, Current-cost, Direction,
        DR-List)
    end
end
```

Algorithm 5: Dynamic routing algorithm

Following the depth-first graph search principle the ITS-DFS traverses the graph and finds the smallest cost of reaching each intersection. The procedure retains information on the oNRB that was used at the start and is therefore able to associate the lowest cost route with the direction that hast to be taken on the intersection from which the search originated. Such a process is illustrated in Figure 3.6.

In this example the DR mechanism is examined from the point of view of vehicles approaching *is4* through *r54* link. Assuming that all edges have equal non-zero weighting the DR mechanism would advise the vehicle to take *r42* link to reach intersections *is1* and *is2* and use *r43* link to reach *is3* and *is5.* It is worth noting that due to the fact internal representation of *is4* does not allow U-turns (see Figure



From r54:

Destination	Direction	Path
is1	r42	is4->is2->is1
is2	r42	is4->is2
is3	r43	is4->is3
is5	r43	is4->is3->is2 ->is4->is5

Figure 3.6: An example of Dynamic Routing graph search result.

3.5) the route to *is5* leads through *is3*, *is2* and *is4*.

3.7 Conclusions

This chapter introduced the traffic management methods used in this work. Among them were four intersection control algorithms (ICA), a system optimising how vehicles approach intersection and a dynamic vehicle routing method. First two ICA formed a benchmark that is used to evaluate the Two-Step technique. The novel Two-Step traffic management technique enables the intersection controller to react to demand in an adaptive manner and advise vehicles on the intersection approach trajectory. It is based on a pressure based intersection management algorithm that takes advantage of the capabilities of ITS sensors.

It was realised that all the above mentioned techniques access to various data, either about the current traffic situation or regarding the configuration of the road network. Intersection control algorithms need to be able to associate the lane sensors with traffic stages numbers, the Two-Step method additionally needs to know the geographical locations of the sensors to be able to calculate the prediction horizon. The dynamic routing component needs to build the road network graph based on information provided by the intersections.

The ITS-Cloud distributed processing platform described in next chapter was created to perform such data management and provide an execution environment for the ITS traffic management applications introduced in this chapter.

Chapter 4

The *ITS-Cloud* platform

This chapter describes the ITS-Cloud, a cloud computing based distributed processing and data management platform which is able to support the traffic management solutions developed in Chapter 3. The chapter has the following sections:

- The first section introduces the ITS-Cloud platform and points out its key elements.
- The second section describes how grid and cloud service models have been incorporated in the ITS-Cloud.
- The third section introduces the new multi-dynamic service concept built by combining selected features of the service types described in the previous section.
- The fourth section describes the Service Discovery System (SDS) for service discovery and allocation within the ITS-Cloud.
- The fifth section describes the SDS mechanism exploitation within the ITS-Cloud to increase reliability and fault tolerance.
- The sixth section describes the cloud interface, a set of functions that make the mechanisms mentioned in the previous section transparent to the user and allow user code to interact with the ITS-Cloud.

- The seventh section discusses the communication issue of distributed computing systems and describes the messaging system used by the ITS-Cloud.
- The conclusions are presented in the last section.

4.1 The ITS-Cloud platform

The ITS-Cloud is a distributed computing platform, based on cloud and grid computing principles, designed and implemented as part of this work. Its main purpose is to host all the ITS applications that form the Cloud based Traffic Management System (CTMS) and manage communication between them.

The concept of using cloud computing principles to construct a traffic control and management system has been described and published by the author and co-writers in [128]. The ITS-Cloud platform is composed of:

- **Services**

  Services are the basic building blocks of service oriented architecture (SoA) systems such as the ITS-Cloud. Services are independent software components designed to perform specific functions in the system. All the processing in ITS-Cloud is conducted by means of coordinated interaction between software services.
- **Resources**

  In ITS-Cloud resources are service containers. Their task is to instantiate other services and manage them throughout their life time.
- **Service Discovery System**

  The Service Discovery System (SDS) keeps a record of all service instances that exist in the system and are available to users. The SDS is also responsible for allocating new services on appropriate resources when requested.

4.2 Service types

There are three types of services available in the ITS-Cloud: static, dynamic and multi-dynamic. The first two types are based on standard cloud and grid computing approaches and the third was introduced in this work due to specific requirements of ITS applications.

Services and resources available on the grid systems are usually considered as static. This means that the services are shared between all users and exist in the system regardless whether they are being used or not. It is the responsibility of the service to distinguish between different users. It can be assumed that such static services are available throughout the lifetime (execution time) of a user application [118].

In cloud computing based infrastructures service instances are created (allocated) on demand, usually to serve only one client exclusively. After the client disconnects, the service instance is de-allocated (destroyed). Such an approach benefits from increased security through user isolation and allows using simple single user services [119]. The disadvantage is that a complex service management layer is necessary to allocate and manage such services. The differences between the cloud and grid service models are illustrated in Figure 4.1.

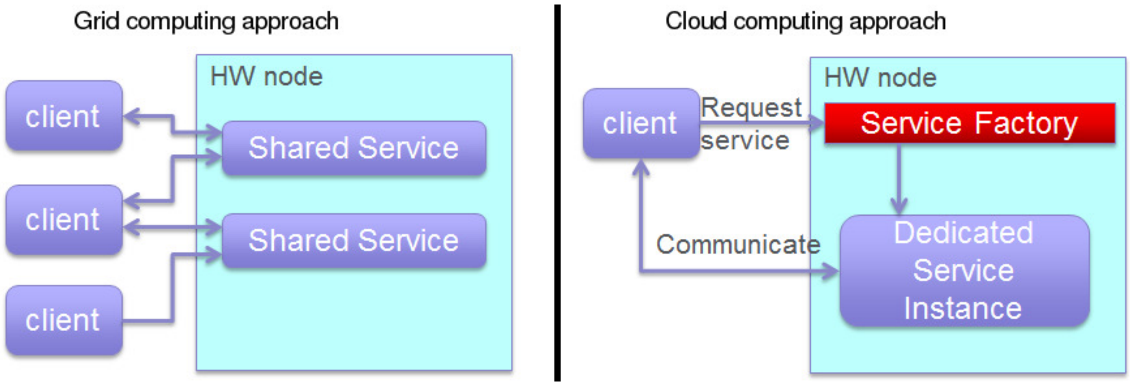

Grid computing approach
HW node
client
client
client
Shared Service
Shared Service
Cloud computing approach
HW node
client
Request service
Service Factory
Communicate
Dedicated Service Instance

Figure 4.1: Comparison of cloud and grid service models.

Based on the cloud and grid variants of software services, the two base service types have been identified for use with the ITS-Cloud namely static and dynamic. Static services are based on a grid computing service model. They can be shared between multiple users and are always present in the system. Dynamic services follow the cloud computing service model. They are allocated on demand for exclusive use when a user requests them. In ITS-Cloud the resources are responsible for allocating and managing dynamic services.

Figure 4.2 provides an overview of the ITS-Cloud system and its components. The base system is composed of the SDS and several resources. The dynamic and multi-dynamic services are allocated on the resources and static services can be based anywhere as long as they are registered with the SDS.

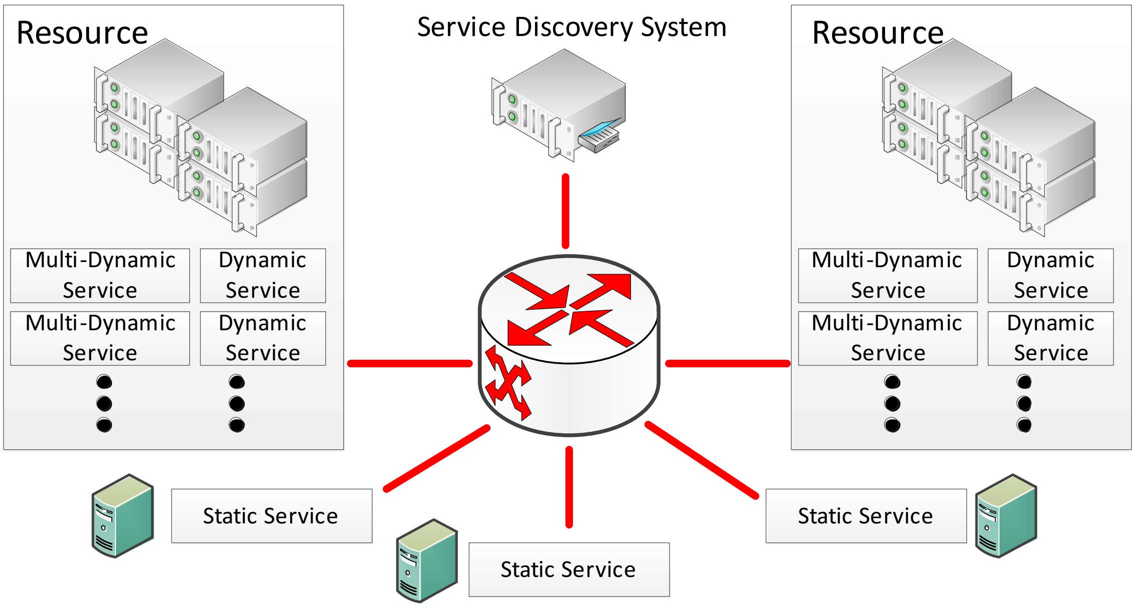

Resource
Service Discovery System
Resource
Multi-Dynamic Service
Dynamic Service
Multi-Dynamic Service
Dynamic Service
Multi-Dynamic Service
Dynamic Service
Multi-Dynamic Service
Dynamic Service
Static Service
Static Service
Static Service

Figure 4.2: Components and services of the ITS-Cloud system. Dynamic and multi-dynamic services are contained by the resources. Static services can be located anywhere in the system. All services are registered with the SDS.

4.3 Multi-dynamic service allocation

Previous section introduced static and dynamic service types, commonly used in grid and cloud computing environments. This section introduces a new ITS specific service type termed multi-dynamic service.

Throughout the design process of the CTMS and its applications it was determined that a new type of service will need to be implemented in the ITS-Cloud. The new service type was required to handle multiple users, like a static grid service, but should still be allocated dynamically, similarly to dynamic cloud services.

When a client wishes to invoke a multi-dynamic service it sends a request to the Service Discovery System (SDS), which then determines if such service exists in the cloud. If the requested service is not present the SDS endeavours to create it, as illustrated in Figure 4.3. A suitable resource for hosting the new service instance is selected, out of the resources available in the system, and requested to allocate the

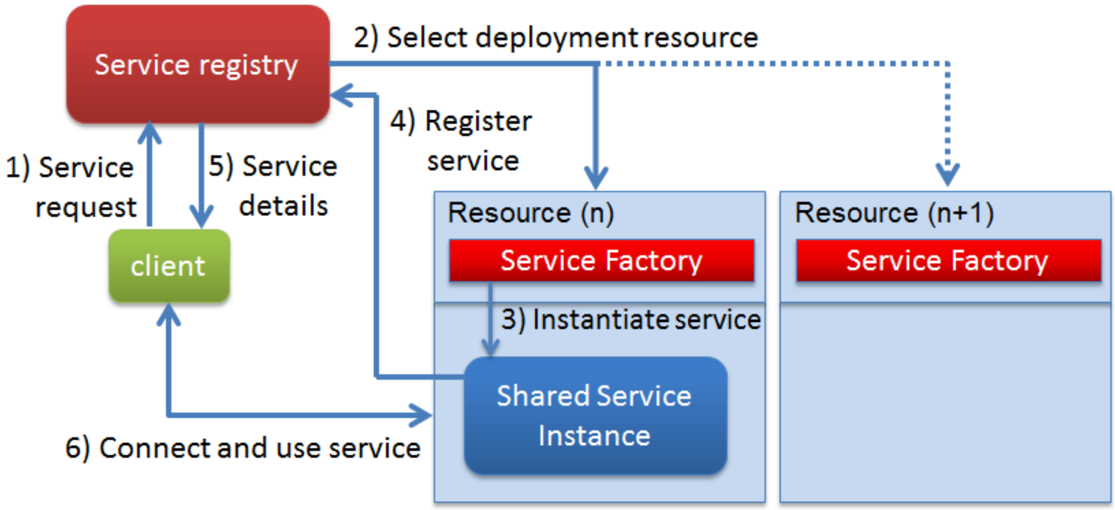

Service registry
2) Select deployment resource
4) Register service
1) Service request
5) Service details
client
Resource (n)
Service Factory
3) Instantiate service
Shared Service Instance
6) Connect and use service
Resource (n+1)
Service Factory

Figure 4.3: Multi-dynamic service allocation: 1 - Client requests a service from SDS. 2 - SDS knowing that such service does not exist in the system selects a best resource to allocate it on. 3 - Resource creates service instance. 4 - Service registers with the SDS. 5 - User is provided with the service handle. 6 - The user establishes direct connection to the service.

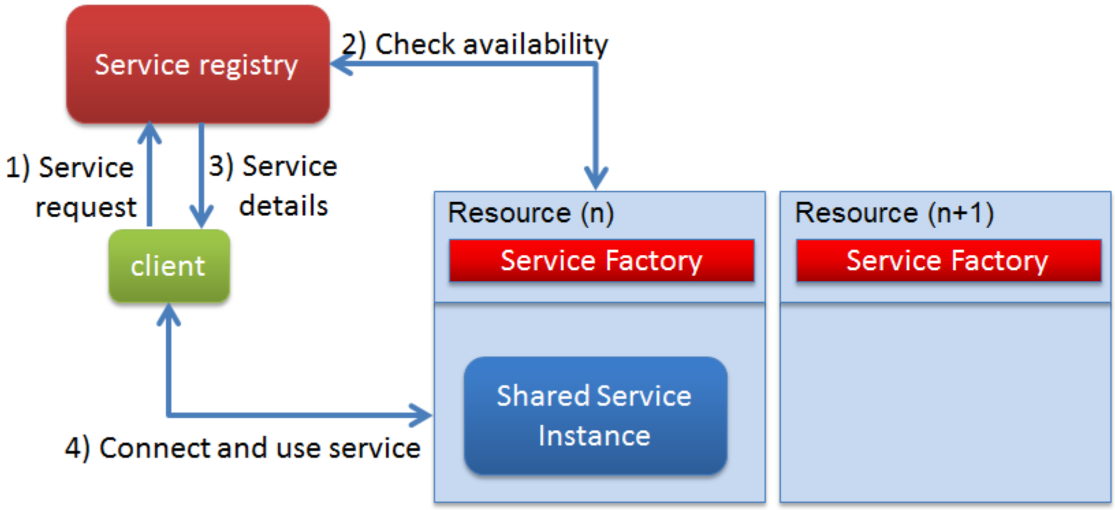

Service registry
2) Check availability
1) Service request
3) Service details
client
Resource (n)
Service Factory
Resource (n+1)
Service Factory
Shared Service Instance
4) Connect and use service

Figure 4.4: Multi-dynamic service access: 1 - Client requests a service from SDS. 2 - Knowing that such service already exists in the system SDS ensures its availability. 3 - User is provided with the service handle. 4 - The user establishes direct connection to the service.

desired service instance.

Once the service is allocated the resource provides the SDS with a service handle, which is then forwarded to the user. The service handle contains all information necessary for the user to connect to the service and communicate with it. In ITS-Cloud the handle consists of the network IP address of the computer hosting the service and TCP port number under which the service is available. The service handle is stored in the SDS and is provided to other users requesting the same service in future as demonstrated in Figure 4.4.

In cases where the service is not available in the system and needs to be allocated the SDS is tasked with selecting the best resource to host the new service.

4.4 Service discovery and deployment

The Service Discovery System (SDS) handles service discovery and allocation processes of dynamic and multi-dynamic services on the ITS-Cloud platform.

4.4.1 Service discovery

The ITS-Cloud platform relies on a centralised SDS to provide service discovery functions. Centralised service discovery systems are simple to implement but can become a bottleneck that can affect the performance of the entire system.

Decentralised service discovery systems offer much greater flexibility. They are often based on heuristic or genetic algorithms which might not always yield the optimal result, but are much better suited for very large systems than a centralised SDS [129].

The relatively small scale of the ITS-Cloud system warrants use of a centralised service discovery system. It is realised as a service database maintained within the SDS. The following information are stored by the SDS about each service in the ITS-Cloud:

- **Identifier**
  A unique numeric value being the primary key of the service record in the database.
- **Service name**
  A character string uniquely identifying the service instance in the system. The SDS will prevent the creation of two services with the same name.
- **Service class**
  The allocation class of the service. It can either be static, dynamic or multi-dynamic.
- **Service type**
  The type of the service identifying its function in the ITS-Cloud, for example resource, sensor service, intersection management service.
- **Service handles**
  A service handle contains information that allows users connect to it. In ITS-cloud a service handle consists of the IP address of the host and the TCP port number the service it is using. There can be several handles stored in

this field if the service has been duplicated (see Subsection 4.4.3).

- **Service capabilities and information (SCI)**

  This field is a list of character string to character string mappings providing a universal method of storing additional information about the service. The first string is the key used to access the second string which contains the information.

- **Timestamp**

  The timestamp states the age of the service database entry. The services that fail to regularly update their status will be removed from the service list.

The clients can query the SDS for services using any combination of name, service class, service type and service information. There are two types of queries available:

- **Simple query or allocation request**

  The simple query requires the user to specify a service characterised by name and optionally by class and type. The query can either result in a single service being found or allocated or in a query failure.

- **Complex query**

  Complex queries can return a list of services matching the query criteria. Users can search for services using any combination of service class, type or SCI. This query type exploits relations between services and ITS concepts provided by the SCI of each service. Examples of complex inquiries include: accessing all sensors from a specific road or lane, obtaining a list of adjacent intersections from a given intersection or determining the relationship between signalling stage numbers and lane sensors.

A successful query will result with the user being provided with the service name, class, type and a handle to allow direct connection.

4.4.2 Service deployment

This subsection investigates the method used by the SDS to determine the most suitable resource to deploy a new service upon.

In order to deploy a new service the SDS has to be aware of the location of the resources and the capabilities of the computers they are hosted on. In the ITS-Cloud resources use the same service model as any other service in the system, therefore they can be found in the SDS service database. Resources provide, in their SCI mappings, information about the host machine they are deployed on. Such information includes a number of processing cores, available RAM memory and the amount of services already allocated on the resource.

The service should be allocated taking into account user expectations and the internal allocation goals of the cloud system. From the user point of view the service should be allocated in such way that all the computational power and memory requirements are fulfilled, latency is minimised and there is sufficient bandwidth to the service. The cloud system aims to prevent overloading of resources and saturating the network links.

Latency becomes an issue with very large scale distributed systems with users, resources and services spread throughout the world. The ITS-Cloud system in its current form was designed to operate in urban environment and using a good quality network connection, therefore the issues of bandwidth and latency in service allocation were dismissed in the current version of SDS.

The service deployment algorithm uses the resource suitability index R_s to determine which resource should be used to deploy a new service. It is aimed to spread the computational load evenly though the system. The index is calculated as follows:

$$R_s = \frac{R_{CPU}}{R_{count}} \tag{4.1}$$

Where R_{count} is the amount of services deployed on the resource and R_{CPU} is the amount of available processing cores. The index is set to zero if the resource does not fulfil all the requirements specified by the service.

The potential resource list is then sorted in descending order of the suitability index and the SDS attempts to allocate the service on the most suitable resource. If the selected resource fails to allocate the service it will be removed from the available resources list. It will be therefore required to re-register with the SDS in order to assure that the resource is still operational.

4.4.3 Service migration and duplication

The service migration mechanism relocates a service from one resource to another. In many distributed processing systems services are relocated with the aim to optimise performance of the system or when their current host is removed from the processing pool. Such a migration process is transparent to the service user.

Services in the ITS-Cloud can be migrated when the resource hosting them is shutting down. It is done as follows:

1. The resource that is about to shut down is removed from the SDS resource list to prevent the allocation of new services.
2. The SDS is used to allocate a new service instance on a different resource.
3. The current service context (the internal state of the service) is copied to the newly allocated service instance.
4. User handle is updated do point to the new service instance.
5. The old service is de-allocated.
6. Points 2 to 5 are repeated until all the services are migrated.
7. The resource is shut down.

Service duplication works using the same principles as service migration, however the original service is not de-allocated. A record of a duplicated service in SDS holds handles for each duplicates of the service. The service duplication mechanism is used in the ITS-Cloud to improve reliability of the traffic management system.

4.5 Reliability, fault tolerance and redundancy

The previous section described the ways service allocation, migration and duplication are handled in the ITS-Cloud system. This section describes how such mechanisms can be used to improve reliability of the services provided by the system.

Service migration can protect against resource failure only if the service can be moved off the failing resource in time. Such necessity may arise when the computer hosting a resource switches to battery power due to a power failure. In such case the time to failure is determined by the battery life, to ensure that there is sufficient time to safely migrate all the services and shut down the resource.

If a resource fails without prior warning all services it contained are lost. While traffic management systems, including CTMS, are equipped with emergency procedures for such occasions, such service loss will affect the user. When a traffic management system is hosted on the platform such failure would lead to temporary degradation in traffic management performance.

In order to prevent unexpected loss of resource from affecting the operations of the system the ITS-Cloud always maintains two copies of the key services on different resources. The system makes sure that all the duplicates are synchronised (kept in the same state) so when one fails the other can immediately take over without affecting the user.

If a non-duplicated service is lost or in an unlikely case of simultaneous loss of all service duplicates the ITS-Cloud will re-create the lost services on different resources, however their state will be lost. In such situations the user will be notified and will have to handle the situation accordingly.

In order to provide such functionality to the user in a transparent way it is necessary to introduce an additional communication layer between the user code and the services. Such a layer is referred to as the cloud interface and is discussed in the following section.

4.6 The cloud interface and user interaction

This section describes the service duplication and reallocation mechanisms from the user perspective as well as the user communication with the ITS-Cloud.

Some of the ITS-Cloud features, such as the service duplication, require some engagement from the user application side. The cloud interface is a set of library functions written in *Java* that handles all the communication between the user and

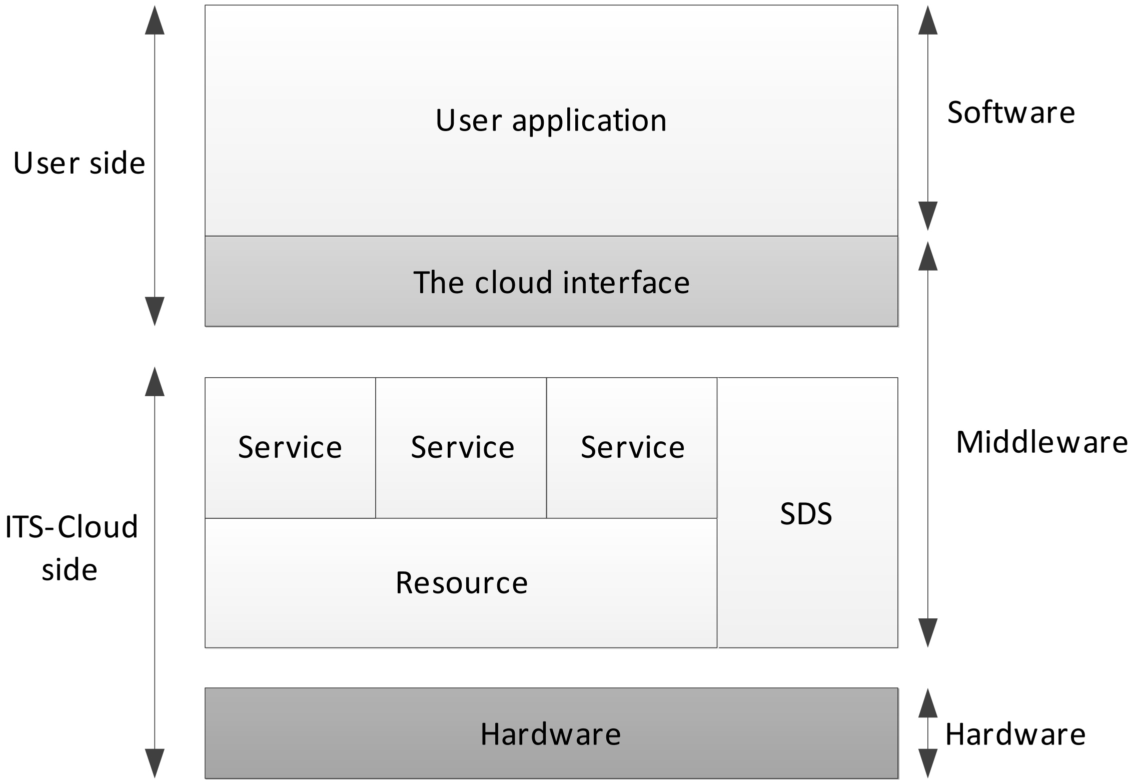

User side
ITS-Cloud side
User application
The cloud interface
Service
Service
Service
SDS
Resource
Hardware
Software
Middleware
Hardware

Figure 4.5: Layer model of the ITS-Cloud platform.

the ITS-Cloud system. Figure 4.5 illustrates the layer model of the ITS-Cloud. The concept of middleware was developed alongside with the concept of grid computing [115, 118]. The middleware is an additional (middle) layer between the software and the hardware that organises access to hardware in grid systems. Following that principle the cloud interface can be categorised as middleware that organises the way users access the computational power of the ITS-Cloud system.

The cloud interface provides the user with an application programming interface (API) which gives the user access to the following functions:

- **Management of service connections**

  This function enables the user to instruct the cloud interface to connect to or disconnect from a service. If a connection to a new service is requested, the middleware will seek service handle(s) from the SDS and establish a communication session with the desired service.
- **Communication with the services**

  The cloud interface handles the communication between the user and the service. It provides functions for both sending and receiving messages from the service. When the service has duplicates it makes sure that all the user input is delivered to all the duplicates in order to keep their state consistent. The service responses are monitored to ensure that the reply from only one of the duplicates is delivered to the client.

The following section describes the message exchange format required to support the aforementioned functions.

4.7 Messaging

Previous section described the user-service interaction mechanism and described the role of the cloud interface. This section discusses the communication aspect of distributed computing systems and describes the messaging format used in the ITS-Cloud.

Operation of all cloud systems is based on the interaction between the users and services. In most distributed systems the services and users are roles, therefore a service can be also considered a user if it invokes another service.

Services interact with each other by exchanging messages. In generic clouds the format used for messaging is usually an XML derivative such as SOAP [130]. XML based messaging formats offer great flexibility as the message contains the payload as well as tags describing it. The way any client should interact with services can be defined using WSDL, another XML based language. Using WSDL the client gains knowledge on the network location of the service and methods to formulate requests in SOAP and learns of the service's capabilities. The flexibility of SOAP is offset by large processing and transmission overheads.

```
POST /InStock HTTP/1.1
Host: www.maths.org
Content−Type: application/soap+xml; charset=utf−8
Content−Length: nnn
<?xml version=”1.0”?>
<soap:Envelope
xmlns:soap=”http://www.w3.org/2001/12/soap−envelope”
soap:encodingStyle=”http://www.w3.org/2001/12/soap−encoding”>
<soap:Body xmlns:m=”http://www.maths.org/multiplicator”>
   <m:Multiply>
      <m:Parameter1>5</m:Parameter1>
      <m:Parameter2>3</m:Parameter2>
   </m:Multiply>
</soap:Body>
</soap:Envelope>
```

Listing 4.1: SOAP Example

Listing 4.1 shows an example of a basic SOAP request to a *multiplicator* service hosted on *www.maths.org* server. The requests contain two parameters that form the input to the service.

Assuming that the example SOAP message was encoded using ASCII character set (1 byte per character) the above example would be at least 427 bytes long (depending on the parameter length) and the response would be of a similar magnitude. SOAP messages usually use UTF-8 character encoding which is a variable width character encoding and would introduce even more overhead. A minimalistic interaction scheme for this service would involve sending two parameters in plain binary. Using IEEE 754 double precision floating point format such message would be 16 bytes long.

The majority of communication in ITS-Cloud is based on high frequency exchange of small amounts of data, therefore such overheads were deemed unacceptable. In order to address that issue the messaging system used in ITS-Cloud deviates from the generic SOAP/XML and uses a custom messaging format and the *Java* serialization mechanism instead.

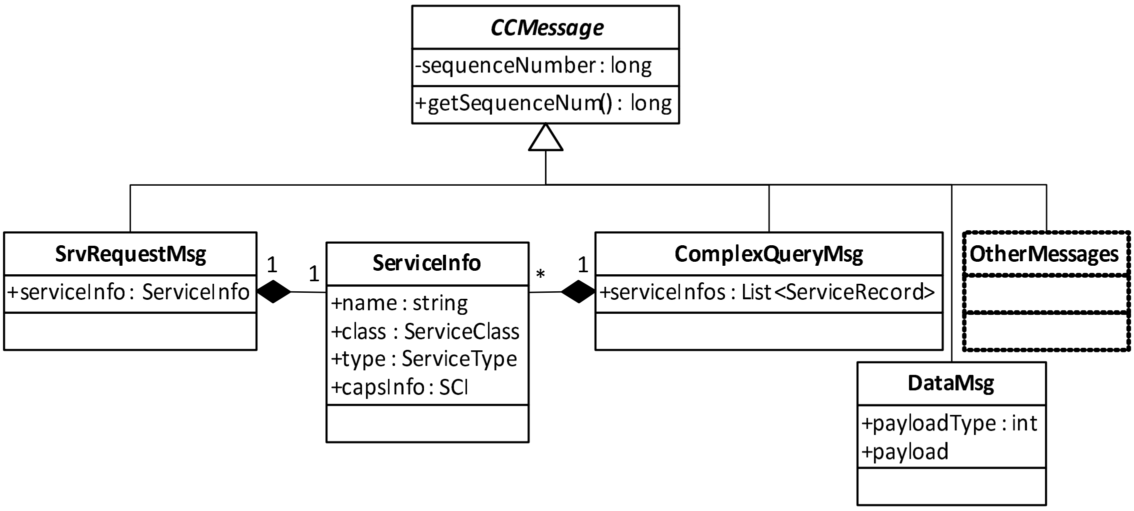

CCMessage
-sequenceNumber : long
+getSequenceNum() : long
SrvRequestMsg
+serviceInfo : ServiceInfo
1
1
ServiceInfo
+name : string
+class : ServiceClass
+type : ServiceType
+capsInfo : SCI
*
1
ComplexQueryMsg
+serviceInfos : List<ServiceRecord>
OtherMessages
DataMsg
+payloadType : int
+payload

Figure 4.6: Class diagram of the message hierarchy in the ITS-Cloud.

Serialization allows converting an object into a byte stream (serializing) that can be sent over a network connection and converted back to an object representation (de-serialized) on the other side [131]. This allows skipping the time consuming parsing of a SOAP message and has much less transmission overheads.

Figure 4.6 illustrates a selection of messages used in the ITS-Cloud system. All the messages used in the system are based on a generic *CCmessage* object and inherit the sequence number parameter form it. The sequence number is used by the ITS-Cloud services and the cloud interface on the user side to relate responses to queries, to detect message losses and in the service duplication scheme to ensure that the client receives only one response.

Each service type has its own range of messages that are used to interact with it. Such messages have to be defined before any communication with the service can occur.

4.8 Conclusions

This chapter introduced the ITS-Cloud, a service oriented distributed computing platform designed to host ITS applications. A new multi-dynamic class of services has been introduced that builds upon cloud and grid computing service models.

Owing to the fact the ITS-Cloud was designed with a purpose of hosting a traffic management system in mind, the design and implementation of some of the platform's components diverge from their definition and implementation in other cloud computing systems. Those deviations were made mainly due to performance and reliability concerns, and to keep the ITS-Cloud platform relatively simple. Drawbacks resulting from such specialisation of the developed platform include mainly reduced messaging flexibility due to the messaging system used, and a possible bottleneck due to a centralised SDS being used. The advantages include simplified deployment and cross-platform portability.

The next chapter introduces the Cloud based Traffic Management System (CTMS). CTMS is a collection of services deployed on the ITS-Cloud.

Chapter 5

The Cloud based Traffic Management System (CTMS)

The previous chapter discussed the ITS-Cloud, a distributed processing platform for ITS applications. This chapters introduces several traffic management applications that are deployed on the ITS-Cloud as services and form the Cloud based Traffic Management System (CTMS).

- The first section provides an overview of the CTMS, defines the aims of the system and outlines its main tasks. It also provides a brief overview of the system components.
- The second section outlines the approach to traffic management and identifies the areas of traffic management handled by the CTMS.
- The third section describes how traffic situation images, necessary to perform any kind of traffic management, are created.
- The fourth section describes how the intersection control algorithms introduced in Chapter 3 are used in the CTMS.
- The fifth section describes the Meso Scale Prediction Service (MeSPS).
- The sixth section describes the cloud interface, a set of functions that make

the mechanisms mentioned in the previous section transparent to the user and allows user code to interact with the ITS-Cloud.

- The seventh section discusses the communication issues associated with distributed computing systems and describes the messaging system used by the ITS-Cloud.
- The conclusions are presented in the last section.

5.1 CTMS overview

The CTMS aims to provide:

- **Intelligent traffic flow management**

  The main goal of the CTMS is to manage the traffic flow with aim to improve road utilisation, reduce average journey times and energy consumption.
- **Scalability**

  The traffic management system should be fully scalable. It should be possible to add or remove components such as sensors, VMS or even whole intersections from the system without having to shut it down. The impact of such changes on other system components should be minimised.
- **Reliability**

  The system should be able to provide a reliable and uninterruptible service. It should also endeavour to mitigate the consequences of component failure.

The CTMS follows the sample-compute-actuate approach to perform its function. Such approach is illustrated in Figure 5.1 and is divided into three interrelated tasks:

- **Traffic data gathering**

  In order to perform any kind of traffic control the traffic situation on roads has to be constantly monitored. The CTMS is tasked with gathering traffic data from the road network and maintaining an up to date traffic situation image. The traffic data is gathered from various types of sensors and the appropriately equipped vehicles can also contribute to the creation of the situation image by providing their location by means of wireless communication.

- **Traffic control decisions generation**

  Having created the traffic situation image the system has to decide what to do in order to achieve its optimisation goals. Such decisions are made using the traffic management methods described in Chapter 3.

- **Traffic actuation**

  The decisions made by the CTMS have to be enforced on the traffic network. This is achieved by controlling the traffic lights and issuing appropriate advices to the vehicles.

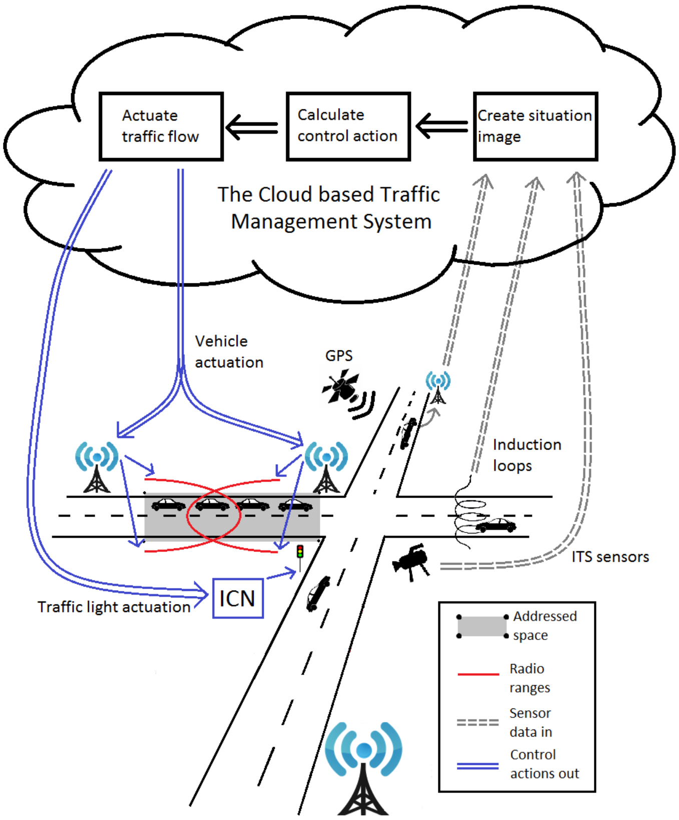

Actuate traffic flow
Calculate control action
Create situation image
The Cloud based Traffic Management System
Vehicle actuation
GPS
Induction loops
ITS sensors
Traffic light actuation
ICN
Addressed space
Radio ranges
Sensor data in
Control actions out

Figure 5.1: Information flow within the CTMS.

The aforementioned tasks are split between the following system components:

- **Sensor Service**

  The Sensor Services (SS) are static services in the ITS-Cloud providing access to traffic data gathered by a sensor. They aims to abstract away the physical implementation of the sensor to enable other CTMS services and ITS-Cloud users with a uniform way of accessing sensor data.
- **Sign Control Service**

  The Sign Control Service (SCS) is responsible for managing variable message signs (VMSs). Such service enables its users to adjust the speed limit on the link where the the VMS is located. VMS can also be used to relay dynamic routing information. Similarly to SS this service is of a static type.
- **Radio Communication Service**

  The Radio Communication Services (RCS) are static services designed to manage an infrastructure node of the vehicle-to-infrastructure (V2I) communication system. They can be used to relay important traffic information and issue advices to V2I-capable vehicles. They can be also act as sensor services for smart vehicles that broadcast their position through V2I.
- **Intersection Control Service**

  Each intersection in CTMS is managed by an Intersection Control Service (ICS). The ICS use SS to construct local traffic situation images and contain intersection control algorithms (ICA) that are then used to control the traffic lights.
- **Intersection Approach Trajectory Optimisation**

  The Intersection Approach Trajectory optimisation (IATO) has been introduced in Section 3.4 of Chapter 3. It is not a standalone service but a part of the ICS that uses RCS to advise vehicles on the optimal intersection approach trajectory.

- **Micro Scale Prediction Service**
  The Micro Scale Prediction Service (MiSPS) is a dynamic service invoked by the ICS. It aims to provide a detailed short term prediction of the traffic situation in the vicinity of the intersection. Such prediction is then used by the novel Two-Step traffic management method described in Section 3.3 of Chapter 3.
- **Meso Scale Prediction Service**
  The Meso Scale Prediction Service (MeSPS) is a functionality of the CTMS achieved as a result of cooperation between multiple ICS. The ICS are able to predict the number of vehicles that will be sent towards the next intersection and is able to advise the ICS governing that intersection on the size of the predicted vehicle group and its estimated arrival time.
- **Area Management Service**
  The Area Management Service (AMS) is responsible for monitoring the traffic in the entire area managed by the CTMS and collection of statistical traffic data. It uses the dynamic routing mechanism introduced in Section 3.6 of Chapter 3 to advise the vehicles with the best route to their destination. Such advices are delivered using RCS.

Having introduced the basic components of the CTMS it is possible to move on to the traffic management process itself. The following sections provide more details about the components mentioned above and describe their interactions. The following subsection introduces the traffic management approach taken by the CTMS.

5.2 Approach to traffic management

The previous section introduced the components of the Cloud based Traffic Management System (CTMS). This section provides an overview of how the CTMS approaches the traffic management problem.

The traffic management process in CTMS is carried out on two management planes: the micro scale traffic management occurs at the intersection level and the macro scale traffic management aims to optimise the whole traffic network.

Figure 5.2 illustrates the organisation of the traffic management processes in CTMS. The managed traffic network, either real or simulated, is observed using various sensors. The sensor data is made available in the CTMS through the Sensor Services (SS). The Intersection Control Services (ICS) use SS to access such data and construct a situation image. Each ICS constructs an image relevant to the intersection it is managing. Such an image is then used to perform intersection level

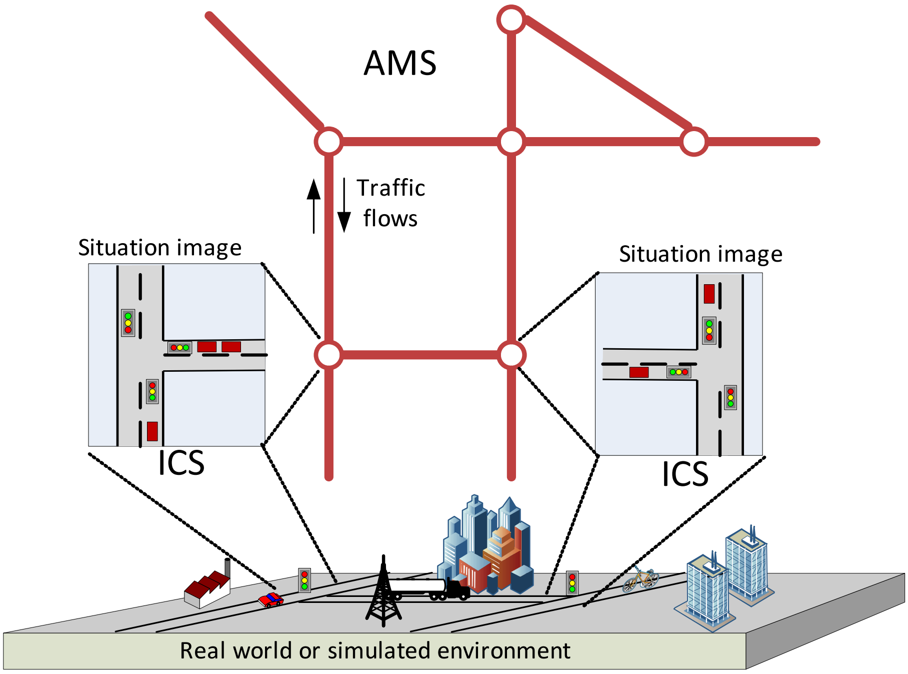


Figure 5.2: Traffic management organisation in CTMS

traffic control. The Area Management Service (AMS) collects statistical traffic data from all the ICS in the managed area.

Local intersection control is referred to as micro scale control as it accounts for individual vehicles and controls the traffic lights accordingly (see Sections 3.1, 3.2 and 3.3 of Chapter 3). The area level traffic control is referred to as the macro scale traffic control. It aims to optimise the traffic flow in the entire network and operates based on aggregated traffic flow data provided by the ICS (see Section 3.6 of Chapter 3).

5.3 Creation of a situation image

The previous section described two planes of traffic control exercised by the CTMS. In order to perform any kind of traffic control up to date information about the traffic situation is required. This section describes how such situation images are constructed.

The creation of detailed situation image at the vicinity of an intersection is necessary in order to employ the adaptive intersection management methods implemented in this work. The situation image comprises static and dynamic components. The static part describes the intersection and the roads that enter it. The dynamic part consists of information on the current locations and speeds of the vehicles. The static part is configured when the ICS initialises and remains invariant thereafter. The dynamic part is constantly updated by obtaining current information from sensor services.

The static road network information is received form the Intersection Control Node (ICN, Section 5.6) upon initialisation of the ICS. It contains all the signalling stages supported by the intersection and the descriptors of the lanes they serve. Such lane descriptors are later associated with appropriate sensors. These sensors will provide the means to build the dynamic situation image.

The creation of the dynamic situation image differs with the available sensor types. Induction loops provide the amount of vehicles accumulated on a given approach (see Subsection 3.1.2 of Chapter 3) and the ILC traffic management technique uses such information directly to perform its task, without the need for creating a situation image.

Data obtained from the ITS sensors (see Subsection 2.2.3 of Chapter 2) contains the vehicle class, location and speed enabling construction of a more detailed situation image. The situation image S_j is created for each lane j entering the intersection separately and is defined by a list of vehicle information:

$$S_j = \{\{V_{1,j}, x_{1,j}, C_{1,j}\}, \{V_{2,j}, x_{2,j}, C_{2,j}\}, ..., \{V_{i,j}, x_{i,j}, C_{i,j}\}\} \tag{5.1}$$

Where the $V_{i,j}$ and $x_{i,j}$ represent the speed and the distance to the stop line of vehicle i on road j, and $C_{i,j}$ represents the vehicle class (e.g. car, bus, truck).

Such information originates from various sources in the CTMS. It is possible that a single vehicle will be reported by multiple data sources. It is therefore necessary to make sure it will not be placed in the situation image more than once. When a data source attempts to add a vehicle to the situation image it has to check if there were prior sightings of that vehicle. If the new measurement of the distance from the stop line $x_{i,j|b}$ places the vehicle within a small distance d from the existing measurement $x_{i,j|a}$ ($|x_{i,j|a} - x_{i,j|b}| < d$) it is assumed that those different data sources observed the same vehicle. The information about such vehicle is therefore integrated as follows:

$$\begin{aligned} V_i &= \frac{V_{i,j|a} + V_{i,j|b}}{2} \\ x_i &= \frac{x_{i,j|a} + x_{i,j|b}}{2} \end{aligned} \tag{5.2}$$

Where subscripts a and b indicate the existing and the new measurements of the vehicle i on lane j. In this work $d = 4m$ was used.

Vehicle class $C_{i,j}$ is not a numerical value and cannot be integrated that way. Therefore the original vehicle class reading remains unchanged, unless the new data has been provided by the vehicle itself thorough V2I wireless communication, in which case the old reading of the vehicle class is replaced.

Figure 5.3 illustrates the situation image creation process. Vehicles 1, 2 and 3 are reported through ITS sensors. Vehicle 2 is reported by both sensors, therefore data integration is necessary. The cooperative vehicle platoon marked as 4 announces itself through the V2I communication. The Meso Scale Prediction Service (MeSPS) can extend the situation image by providing an estimate of approaching vehicles based on information from upstream ICS. Such information allow to include vehicles

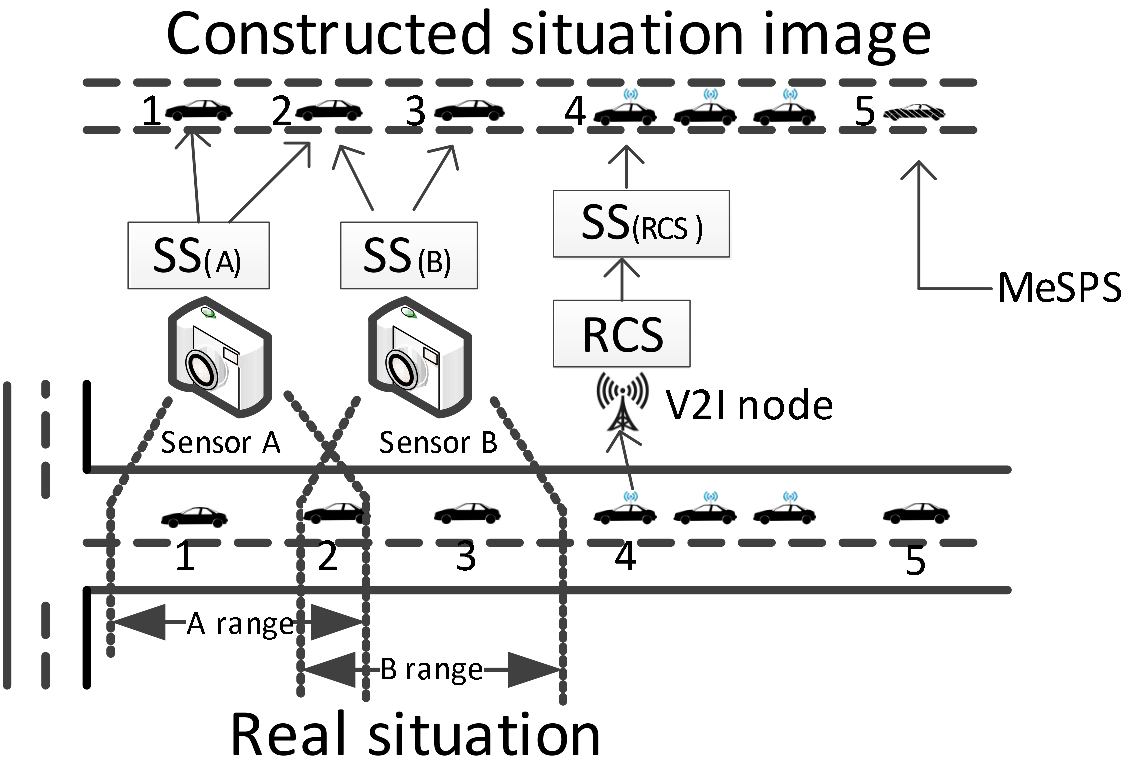

Constructed situation image
1
2
3
4
5
SS(A)
SS(B)
SS(RCS)
RCS
MeSPS
V2I node
Sensor A
Sensor B
1
2
3
4
5
A range
B range
Real situation

Figure 5.3: Construction of a situation image using data from various sources.

in situation image that would be outside of the detection range otherwise (vehicle labelled 5 in Figure 5.3).

5.4 Meso Scale Prediction Service

The previous section described how the CTMS creates traffic situation images using different sources of information. This section describes the mechanism of improving the created situation image using information form adjacent ICS.

Although the Meso Scale Prediction Service (MeSPS) is regarded as a service in the CTMS it is not a dedicated cloud service in ITS-Cloud but a functionality of the ICS that allows them to improve the quality of the constructed situation images. MeSPS allows the ICS to provide advance vehicle flow information to their counterparts down the traffic stream. When a signalling stage is activated, an appropriate message is sent to every downstream ICS reachable from the given stage. Activation of each stage can release a group of vehicles towards more than one possible destinations. This means that at the moment a stage is activated the ICS does not know how many of the queued or approaching vehicles will go in which a direction. A good initial assumption is that a vehicle will choose one of the possible directions from the lane it occupies with the same probability. Therefore probability $P_{out|j}$ of a vehicle choosing one of the outflow lanes can be expressed as:

$$P_{out|j} = \frac{1}{D_j} \tag{5.3}$$

where D_j is the amount of possible destinations from lane j.

The total estimated amount of vehicles N_j that choose a given destination can be written as:

$$\tilde{N}_j = \sum_j \frac{N_i}{C_i} \tag{5.4}$$

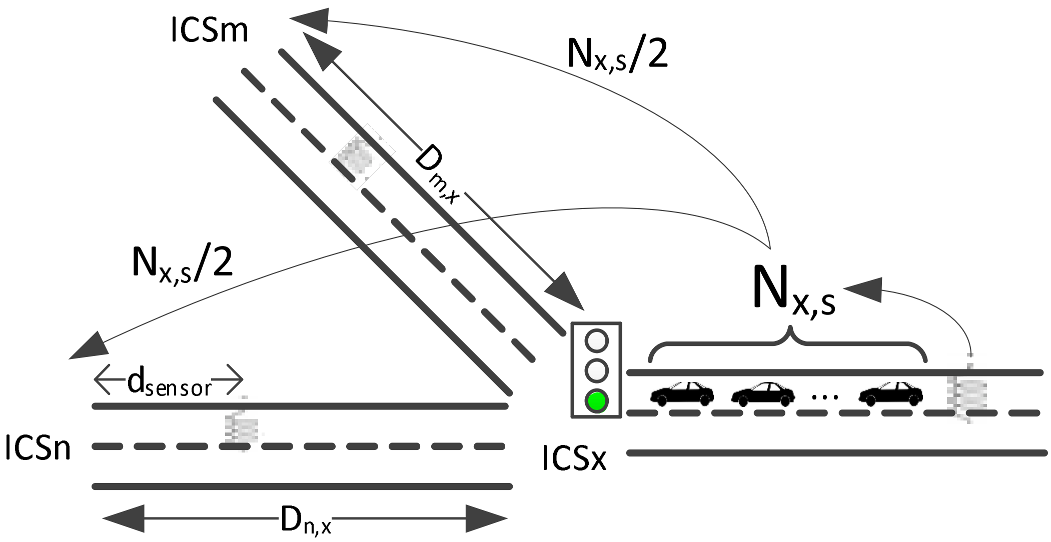


Figure 5.4: MeSPS notifications sent from ICSx to ICSm and ICSn. A group of $N_{id,s}$ vehicles is released from stage s of ICSx. Initially it is assumed that the vehicles will choose a destination with equal probability.

where N_i is the measured amount of vehicles on lane j. The estimated amount of vehicles $\tilde{N}_j$ for each downstream link is sent to the ICS governing the next intersection on each link. Additional information sent include the notifying intersection ID and the associated stage number. An example of such process is illustrated in Figure 5.4. A vehicle platoon is released by intersection governed by ICSx. The vehicles have two possible destinations, they can either head for an intersection governed by ICSn or ICSm. The distances between those intersections and the source are denoted as $D_{n,x}$ and $D_{m,x}$.

The amount of vehicles estimated by the method described above is just an initial guess. The real amount of vehicles can vary depending on the overall traffic network layout, time of day, traffic intensity and possibly other factors as well. Each ICS maintains a weighting parameter $\theta_{id,s}$ for each possible combination of a source (upstream) intersection identifier *id* and a stage number within that intersection s. It is crucial to store the stage number in addition to the source ICS identifier because the flow characteristics for different stages may vary. Based on the information obtained from the upstream ICS and $\theta_{id,s}$ it is possible to estimate the amount of incoming vehicles $\bar{N}_{id,s}$ as follows:

$$\bar{N}_{id,s} = \tilde{N}_{id,s}\theta_{id,s} \tag{5.5}$$

In order to make use of such information and integrate the information on incoming vehicles with the situation image (see Section 5.3), the ICS has to be able to estimate where the lead vehicle is. The distance between the lead vehicle and the stop line is calculated as follows;

$$x_{lead}(t_{id,s}) = D_{id} - V_{max}\eta t_{id,s} \tag{5.6}$$

Where D_{id} is the distance from the intersection managed by the ICS identified by id, V_{max} is the speed limit, η denotes the percentage of maximal speed achieved on average by vehicles on given link and $t_{id,s}$ is the time passed since the platoon was released by the up stream intersection. In this work $\eta = 0.85$ was used. Once $x_{lead}(t_{id})$ is obtained the following $\bar{N}_{id|s}$ vehicles are placed in the situation image behind the leader in spacing $d_{i-1,i}$ defined by the constant time headway policy (see Subsection 6.3.5 of Chapter 6).

As the traffic conditions change it is necessary to keep $\theta_{id|s}$ tuned and up to date at all times. The tuning process is of an iterative and incremental nature. The ICS validates each prediction by comparing it with the measurements taken from the inflow sensors once the predicted vehicle platoon enters the sensing range. The amount of predicted vehicles $\bar{N}_{id|s}$ is compared against the amount of measured vehicles $N_{id|s}$.

In order to perform such validation it is necessary to extract the platoon in question from the general traffic flow. Vehicles are counted as they enter the sensing range of the ITS sensors or pass over an induction loop. It is necessary to start and finish the measurement in an appropriate moments of time. The measurement is started at t_{start} before the lead vehicle estimated arrival time at the point of measurement (denoted d_{sensor} in Figures 5.4 and 5.5) and lasts for t_m until after the estimated platoon is deemed to have passed.

Figure 5.5 illustrates the set-up of such measurement, where a three vehicle platoon (v_1, v_2 and v_3) is released form upstream intersection and travels down the $D_{m,x}$ long road. The time t_{start} where the measurement should start is obtained as follows:

$$t_{start} = \frac{D_{id} - d_{sensor}}{V_{max}\eta} - t_{0,start} \tag{5.7}$$

Where d_{sensor} is the sensing range of the intersection with respect to the stop line (see Figures 5.4 and 5.5), and $t_{0,start}$ is a fixed constant that ensures the measurement is started before the lead vehicle arrives. Value of $t_{0,start} = 4s$ was used in this work.

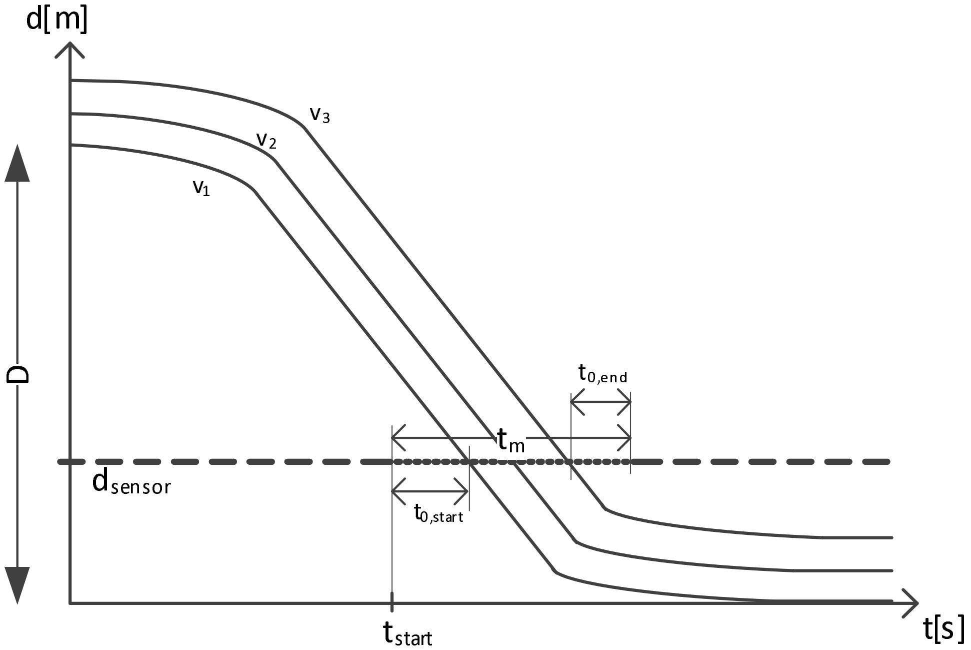


Figure 5.5: Validation measurement set-up. The measurements are carried out using a lane sensor located d_{sensor} from the stop line. The measurement is started at t_{start} and lasts for t_m.

The time of measurement t_m is calculated based on the estimated time it would take the platoon to pass over the measurement point. It is calculated as follows:

$$t_m = t_{0,start} + \bar{N}_{id|s}\tau + t_{0,end} \tag{5.8}$$

Where τ is the time headway, which represents the time gaps between the vehicles. Similarly to $t_{0,start}$ the parameter $t_{0,end}$ defines how long the measurement should continue after the last expected vehicle of the platoon passes the measurement point. Value of $t_{0,end} = 4s$ was used in this work.

After the measurement is finished the amount of counted vehicles $N_{id|s}$ is compared with the amount of predicted vehicles $\bar{N}_{id|s}$. If the values are significantly different the parameter $\theta_{id|s}$ is adjusted as follows:

$$\theta_{id|s} = \theta_{id|s} + \begin{cases} \alpha_{large}, & \bar{V}_{id|s} - V_{id|s} \geq \beta_{large} \\ \alpha_{small}, & \beta_{large} > \bar{V}_{id|s} - V_{id|s} \geq \beta_{small} \\ 0, & |\bar{V}_{id|s} - V_{id|s}| < \beta_{small} \\ -\alpha_{small}, & \bar{V}_{id|s} - V_{id|s} < -\beta_{small} \\ -\alpha_{large}, & \beta_{large} < \bar{V}_{id|s} - V_{id|s} \leq -\beta_{large} \end{cases} \tag{5.9}$$

The scale of adjustment is governed by parameters α_{large} and α_{small} and depends how inaccurate the prediction was. Such inaccuracy is divided into two thresholds β_{large} and β_{small}.

In this work values of $\alpha_{large} = 0.05$, $\alpha_{small} = 0.02$, $\beta_{large} = 8$ and $\beta_{small} = 3$ were used.

Summarising, the MeSPS is a novel feature extending the situation awareness on every ICS beyond the upstream intersections allowing the ICA to schedule the traffic stages more effectively. Using a self tuning mechanism allows the predictor to cope with varying traffic conditions.

5.5 Micro Scale Prediction Service

The previous section introduced a method which aimed to improve ICS traffic situation awareness. This section describes the technique that predicts the evolution of the traffic situation image in near future.

The Micro Scale Prediction Service (MiSPS) is tasked with performing short term predictions of the situation image on intersection approaches, which are then used in the Two-Step traffic optimisation scheme. The situation images is created for each lane as described in Section 5.3. MiSPS predicts traffic situation on each each approach road to the intersection. The initial road situation image is obtained by combining appropriate lane situation images.

MiSPS uses a microscopic traffic simulation engine, from the Traffic Simulator described in Chapter 6, enveloped in a dynamic ITS-Cloud service. Similar to a model predictive controller, it performs a prediction based on a model of the intersection and its surrounding area. This provides the CTMS with a detailed short term predictions of road traffic situations.

Each ICS running the Two-Step intersection control method requests allocation of a MiSPS instance for its exclusive use. The service has to be configured in a similar manner the ICS is configured by ICN on initialisation (see Section 5.3). Static components, such as the road and lane layout, remain invariant between successive invocations of the MiSPS instance and their configuration is part of the service initialisation process.

After the service has been configured it can be invoked by providing it with the starting situation image and specifying the prediction horizon. The current situation image maintained by the ICS is used to as the starting point. The prediction horizon defines how far into the future should the prediction be made, and is calculated and provided to the MiSPS by the Two-Step intersection management algorithm (see Section 3.3 in Chapter 3).

It is safe to assume that the road and lane layout information is accurate, therefore the accuracy of MiSPS prediction depends mainly on two factors: the quality of model representation of individual vehicles in the simulated environment and the accuracy of sensor reading of the initial situation image.

The vehicle model used in the simulation is of sufficient quality to simulate vehicle dynamics necessary to perform traffic management (see Chapter 6), however it was tuned to represent a particular vehicle make and model. In this work the CTMS was evaluated using the Traffic Simulator, where all vehicles are slightly randomised (see Section 6.3 of Chapter 6) variants of the base vehicle model, therefore the inconsistencies between the vehicle models used in the traffic network and those used in MiSPS are not significant.

The use of CTMS with real traffic would require the sensors to classify the observed vehicles, in order to simulate its behaviour using the most appropriate model. It is planned to create a vehicle model for each of the vehicle classes (car, bus, truck etc.) to obtain a prediction of satisfying accuracy.

5.6 Intersection Control

This section describes how the situation image created in the ICS is used to control the traffic.

Each signalled intersection is equipped with an intersection controller that realises a traffic management plan. In CTMS the software of such controller is extended with the ITS-cloud interface, described in Section 4.6 of the previous chapter, and is referred to as the Intersection Control Node (ICN).

Figure 5.6 illustrates the relationship between the CTMS components that perform intersection control. The ICN acts as an ITS-Cloud client (user) and requests allocation of an Intersection Management Service (ICS) and delegates the stage switching decision making process to it. Basic, fixed cycle based, intersection control capability is retained by the ICN in case ICS is not available. The probability of such failure is reduced by using the service duplication mechanism described in Section 4.5 of the previous chapter. There are two duplicates of ICS maintained in different locations in the cloud system and loosing one of them will not affect the intersection management process. The ICS uses the query mechanism provided by the ITS-Cloud (see Subsection 4.4.1 of Chapter 4) to request appropriate sensor

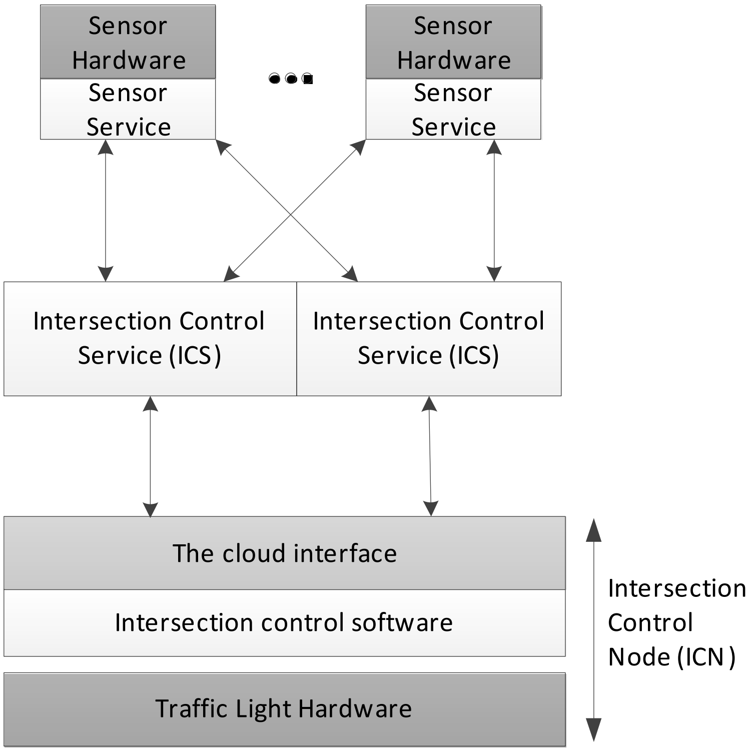

Sensor Hardware
Sensor Service
Sensor Hardware
Sensor Service
Intersection Control Service (ICS)
Intersection Control Service (ICS)
The cloud interface
Intersection control software
Traffic Light Hardware
Intersection Control Node (ICN)

Figure 5.6: Intersection control process in the CTMS. The intersection control node (ICN) requests allocation of an ICS. The ITS-Cloud allocates two duplicates for security. The cloud interface enables communication between the ICN and ICS (see Section 4.6 of Chapter 4).

services that will provide information on traffic in relevant areas, see Figure 5.7.

Figure 5.8 illustrates the data processing and decision making in the Two-Step traffic optimisation method. The situation image constructed based on the sensor data is then used with one of the intersection control algorithms (ICA) to determine

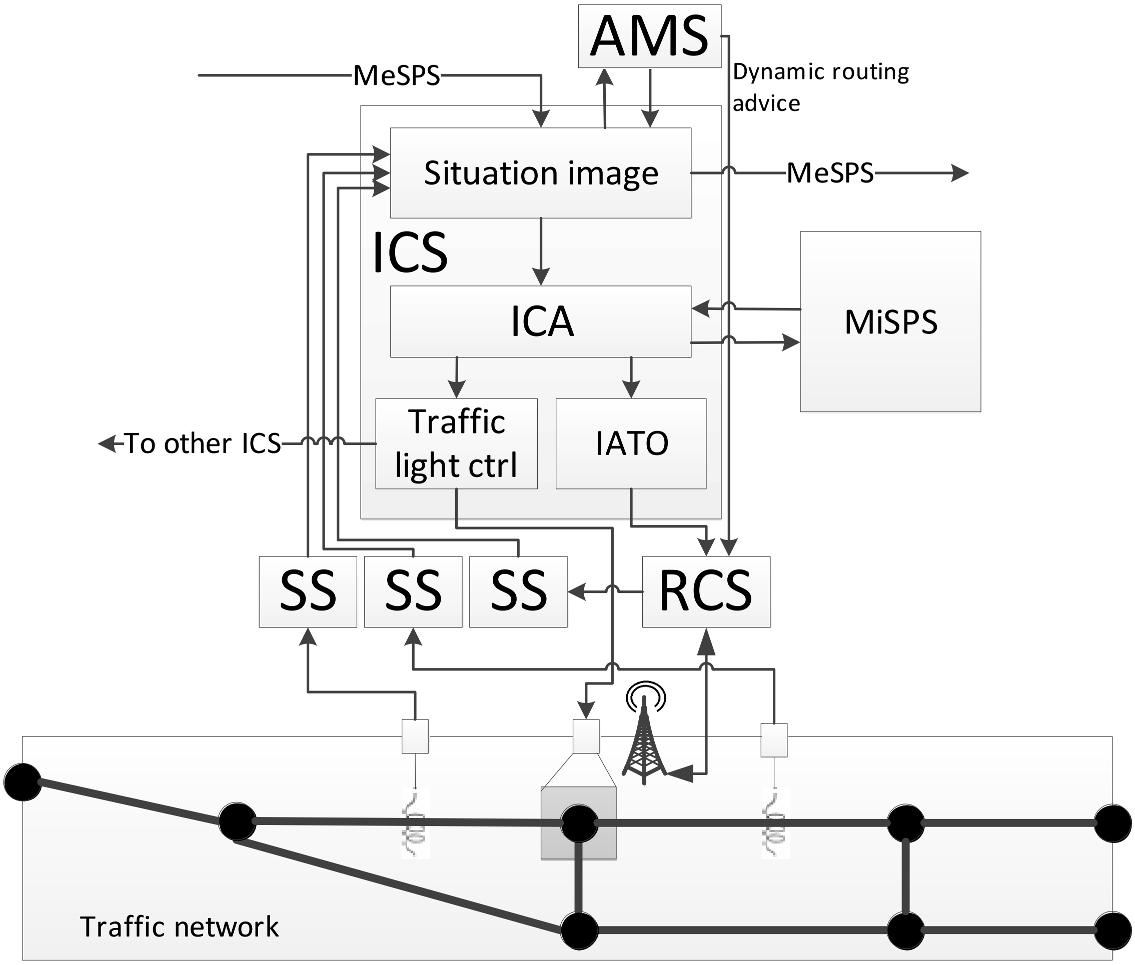

AMS
MeSPS
Dynamic routing advice
Situation image
MeSPS
ICS
ICA
MiSPS
To other ICS
Traffic light ctrl
IATO
SS
SS
SS
RCS
Traffic network

Figure 5.7: Using information provided by the Sensor Services (SS) the Intersection Control Service (ICS) creates a situation image of the traffic in the vicinity of the intersection it manages (see Section 5.3). The Intersection Control Algorithm (ICA) is then used to generate intersection management decision. The novel Two-Step method (see Section 3.3 of Chapter 3) additionally relies on an external Micro Scale Prediction Service (MiSPS) for traffic state prediction. Traffic light switching decision is then provided to the ICN and Intersection Approach Trajectory Optimisation (IATO) is used to control the vehicles.

which signalling stage should be activated. The short term traffic prediction provided by MiSPS is used in the Two-Step ICA.

The Two-Step and FC intersection control algorithms provide advance information on traffic stage change and make it possible to use IATO.

The sensing range of each intersection is limited by the location of the sensors and by the up stream intersections. The ICS instances governing adjacent intersection communicate with each other with aims to increase their respective sensing ranges. This process is governed by the Meso Scale Prediction Service.

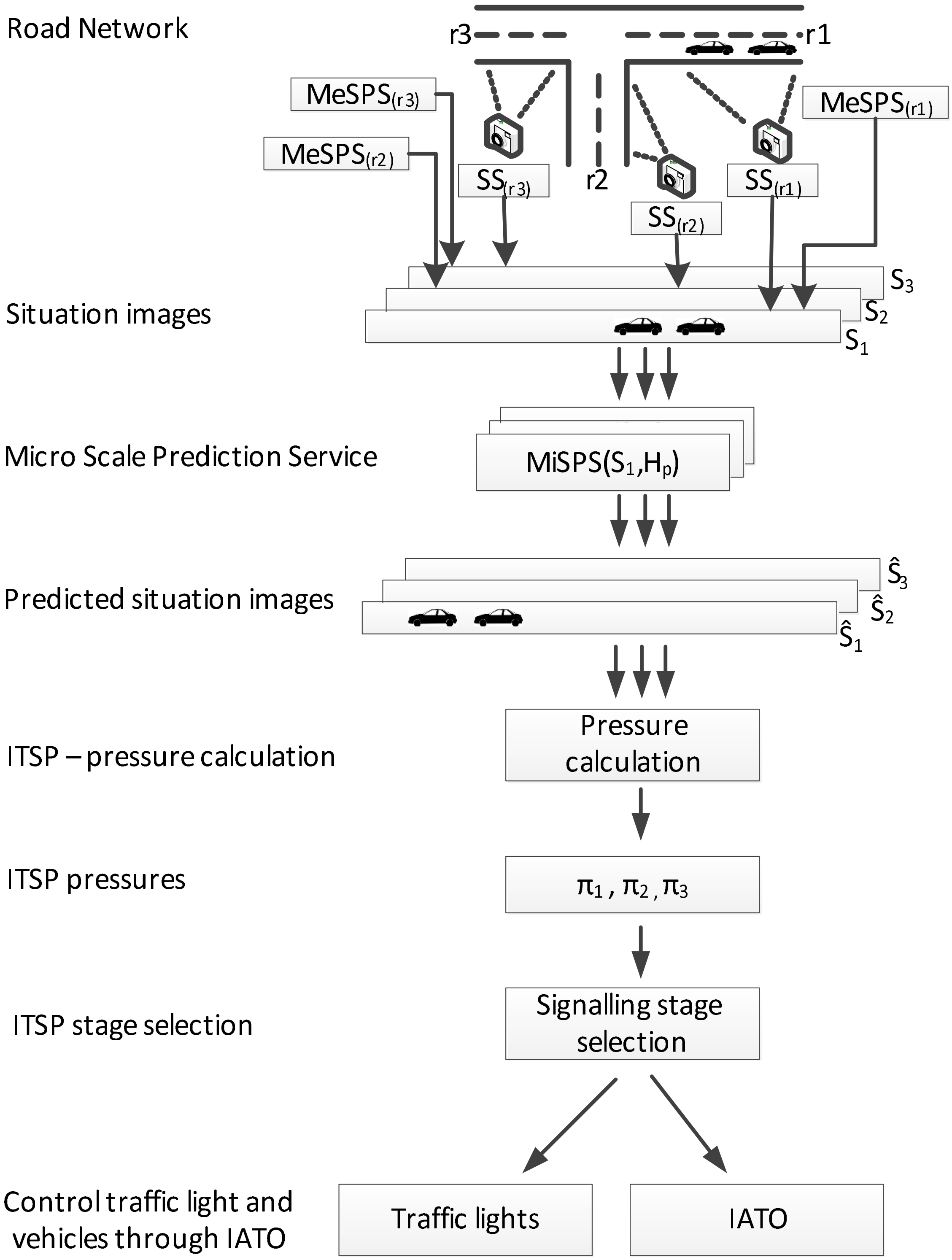

Road Network
r3
r1
r2
MeSPS(r3)
MeSPS(r2)
MeSPS(r1)
SS(r3)
SS(r2)
SS(r1)
Situation images
S3
S2
S1
Micro Scale Prediction Service
MiSPS(S1,Hp)
Predicted situation images
Ŝ3
Ŝ2
Ŝ1
ITSP – pressure calculation
Pressure calculation
ITSP pressures
π1 , π2 , π3
ITSP stage selection
Signalling stage selection
Control traffic light and vehicles through IATO
Traffic lights
IATO

Figure 5.8: Intersection management process using the Two-Step method.

5.7 Intersection Approach Trajectory Optimisation

Section 3.4 of Chapter 3 described how appropriately equipped vehicles can optimise their approach to an intersection. Such process was termed the Intersection Approach Trajectory Optimisation (IATO) in this work. This section describes how the CTMS relays such advice to the relevant vehicles.

An ICA such as the Two-Step or FC are able to provide advance information on future state of the traffic lights. Such information is available as a set of three values: the current active stage number, the next active stage number and the time left before the change occurs. The IATO advice is therefore needed on all lanes belonging to those two stages. The vehicles on the lanes associated with the currently active stage have to be advised that the lights will turn red and the vehicles on lanes associated with the next stage need to know that the lights will be turning green. Before any advice can be sent, IATO has to convert the stage numbers into descriptors identifying each lane in CTMS. Such information is provided by the ICN when the ICS is initialised, therefore IATO being part of ICS can access that easily. The next step is to use the lane descriptors to identify which V2I communication nodes should be used to deliver the advice.

Due to limited ranges of wireless communication in vehicular environment it is very likely that several V2I communication nodes will have to be used in order to cover the entire intended area. The CTMS relies on a form of geographical addressing to determine which V2I nodes need to be used. Geographical addressing aims to deliver information to all recipients in a geographic area. Figure 5.9 illustrates an example where two communication nodes are used to provide coverage of a geographical area.

Such mode of addressing is used for delivering IATO advices. The query mechanism of the ITS-Cloud is used to obtain all the RCS associated with the lane descriptors required and the request to transmit the advice is sent to each of them. It is possible that a RCS covers a wider area than the intended areas/lanes and therefore such communication will be received by vehicles outside of the intended area. It is within the responsibility of the mobile nodes (vehicles) to determine if any received message was meant for them and decide if it should be acted upon. In the simulator described in Chapter 6 it is done by checking if the message was intended to the occupied lane. In the real world a vehicle has to ensure that its position is within the addressed area of the message [132, 133].

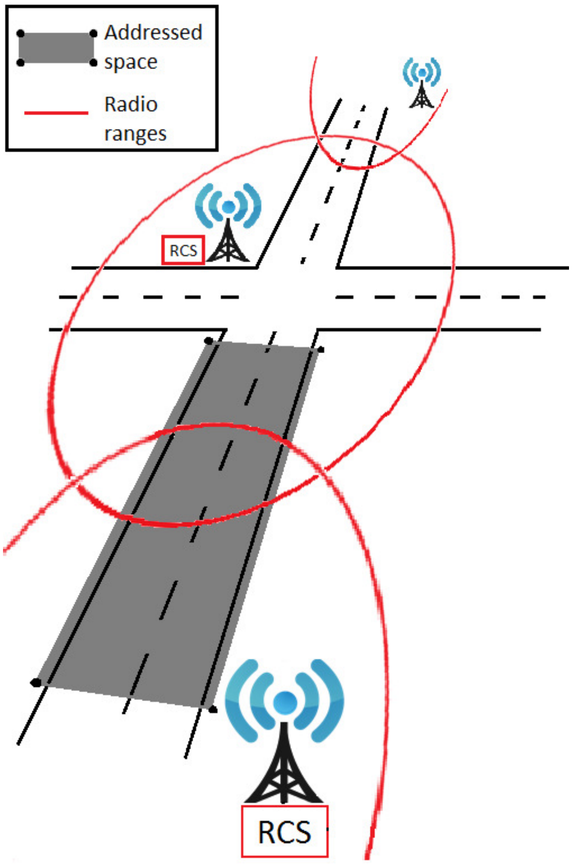


Figure 5.9: Geographical addressing relies on the ITS-Cloud to identify which RCS need to be used to provide the coverage of the desired area (see Subsection 4.4.1 of Chapter 4).

5.8 Failure modes

The previous sections introduced the Cloud based Traffic Management System (CTMS), explained its functions and described its components. This section provides a discussion on possible failure modes of CTMS and its components, together with methods of mitigation.

CTMS was designed with safety in mind. The system accounts for the possibility of component failure and aims to deliver best possible traffic management despite that.

Three major failure modes have been identified in Table 5.1. The first and the worst failure mode assumes total unavailability of traffic management. The second failure mode occurs when the ICN are unable to access ITS-Cloud or if CTMS fails to allocate ICS. The third failure mode deals with single CTMS component failure.

ITS-Cloud resources can host multiple services, therefore even a single resource failure cal result in loss of multiple services thus having a significant impact on the system's performance. The ways of mitigating the consequences of resource failure are discussed in Section 4.5 of Chapter 4.

Loss of dynamic and multi-dynamic services such as the ICS, AMS and MiSPS

Table 5.1: System failure modes

Failure type	Possible causes	Handling
Total system failure	Extended wide area power loss (blackout)	Drivers rely on static traffic signs to safely cross intersections
ITS-Cloud failure or unavailability	Extensive damage to the network, ITS-Cloud registry failure, loss of all resources	ICN fall back to FC intersection control
CTMS component failure	Resource failure, static service failure, partial network failure	varies, see Table 5.2

is always temporary. Provided the ITS-Cloud has at least one functional resource those services will be automatically recreated. Static services such as the SS, SCS and RCS are usually represent a piece of hardware and their failure results in the hardware becoming unavailable in the system. The consequences of CTMS service failure and the methods of mitigation are presented in Table 5.2.

Table 5.2: Service failure consequences

Service	Duplicated	Consequences of failure	Mitigation
ICS	Yes	Loss of intersection control	Loss of one duplicate does not affect the intersection. Loss of both duplicates causes ICN to follow a fixed cycle plan until ICS is re-initialised
AMS	Yes	No area management, no dynamic routing	No effect if one duplicate lost. Loss of dynamic routing capability until new instance of AMS can be initialised.
MiSPS	No	ICS loses prediction capability and is unable to run the Two-Step intersection control method.	Fall back to ITSP adaptive intersection control.
SS	No	Data from the sensor becomes unavailable	Use MeSPS to approximate the traffic situation or revert to FC intersection control.
SCS	No	Inability to control the VMS	Use RCS to relay information to ITS vehicles, no recovery option for normal vehicles.
RCS	No	Inability to communicate with ITS (unavailability of DR and IATO in the affected area).	Use SCS if available.

5.9 Conclusions

This chapter has presented the Cloud based Traffic Management System, a complex urban traffic management solution aiming to optimise traffic flow by means of coordinated cooperation between the vehicle and intersection control approaches.

The system was realised as a collection of cloud services deployed on the ITS-Cloud platform. Such a design ensures scalability and reliability of the system, and provides an abstraction layer between the traffic control algorithm and the sensing/actuating equipment.

The following chapter describes the simulation environment created to provide an evaluation platform for the CTMS.

Chapter 6

The traffic simulator

The previous chapter described the traffic management processes within the Cloud based Traffic Management System.

This chapter presents the microscopic traffic simulation tool intended to cooperate with the CTMS and used to evaluate the traffic management strategies. The chapter starts with a summary of the goals of the simulator followed by the outline of the design in Section 6.2. Section 6.3 describes the vehicle simulation component followed by the approach to simulating wireless communication. Simulation of wireless V2V and V2I communication is discussed in Section 6.4. The traffic network representation and simulation is described in Section 6.5. Sections 6.6 and 6.7 describe traffic simulations set-up and criteria calculated by the simulator to evaluate specific scenarios. Section 6.8 describes the simulation software integration with the ITS-Cloud platform. The chapter is concluded in Section 6.9.

6.1 Goals of the simulator

The main goal of the simulator is to provide a universal and extensible platform for simulating complex traffic conditions on a microscopic scale. The simulator has been designed as a tool to support research on traffic control and its impact on general vehicle flow and individual vehicles behaviour.

The simulator covers most major traffic behaviour such as intersection control, simulation of individual vehicles throttle, brake and dynamic friction. Having such detailed vehicle modelling capabilities is required when trying to determine the impact of traffic control strategies on measures such as road network throughput, travel times, energy consumption and carbon emissions. The simulator should be capable of introducing sufficient amount of randomness into the simulated environment to make the results realistic. The basic assumption is that most vehicles are controlled by a human driver. Therefore the driver behaviour such as reaction latency, limited perception or even variable regard for the traffic rules should be included. Vehicles participating in the simulation should differ with respect to vehicle type, size, mass, engine power and many other metrics. Finally the simulator should possess tools to gather data from the simulation, generate statistics and measure performance with respect to selected performance criteria (see Section 2.1 of Chapter 2).

6.2 Simulator design

The simulator has been designed according to object oriented programming (OOP) design principles [134]. It relies on inheritance and polymorphism to build up the object hierarchy and some of the simulator's functionality. Relationships between object classes presented in this chapter are expressed in the Unified Modelling Language (UML) [135].

The simulator keeps track of all the objects participating in the simulation by using a centralised object registry. The base class for all objects that make the simulated world is called *SimObject*. It allows for objects to be registered with *SimObjectRegistry* and provides interface for destroying objects that are leaving the simulation. The *SimEntity* class is the superclass for all the objects actively participating in the simulation. Active participation means that the object exhibits dynamic behaviour, such as vehicle movement, or has to exhibit situation awareness by interacting with other objects in the simulation.

Figure 6.1 shows the class hierarchy and relationships in the simulator code. The complexity of the design and multitude of objects made it necessary to show only the most representative objects in the figure. Examples of active *SimEn-*

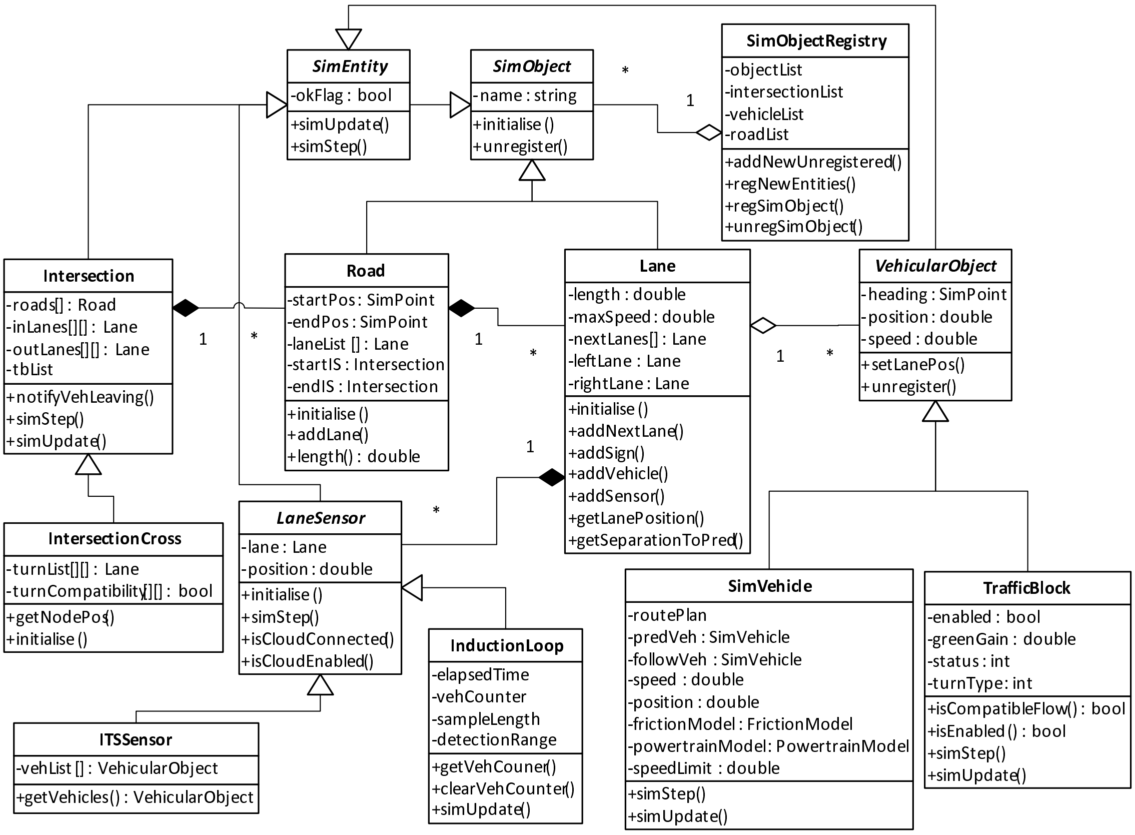

SimEntity
-okFlag : bool
+simUpdate()
+simStep()
SimObject
-name : string
+initialise ()
+unregister()
SimObjectRegistry
-objectList
-intersectionList
-vehicleList
-roadList
+addNewUnregistered()
+regNewEntities()
+regSimObject()
+unregSimObject()
*
1
Intersection
-roads[] : Road
-inLanes[][] : Lane
-outLanes[][] : Lane
-tbList
+notifyVehLeaving()
+simStep()
+simUpdate()
Road
-startPos : SimPoint
-endPos : SimPoint
-laneList [] : Lane
-startIS : Intersection
-endIS : Intersection
+initialise ()
+addLane()
+length() : double
Lane
-length : double
-maxSpeed : double
-nextLanes[] : Lane
-leftLane : Lane
-rightLane : Lane
+initialise ()
+addNextLane()
+addSign()
+addVehicle()
+addSensor()
+getLanePosition()
+getSeparationToPred()
VehicularObject
-heading : SimPoint
-position : double
-speed : double
+setLanePos()
+unregister()
1
*
1
*
1
*
1
*
IntersectionCross
-turnList[][] : Lane
-turnCompatibility[][] : bool
+getNodePos()
+initialise ()
LaneSensor
-lane : Lane
-position : double
+initialise ()
+simStep()
+isCloudConnected()
+isCloudEnabled()
InductionLoop
-elapsedTime
-vehCounter
-sampleLength
-detectionRange
+getVehCouner()
+clearVehCounter()
+simUpdate()
ITSSensor
-vehList [] : VehicularObject
+getVehicles() : VehicularObject
SimVehicle
-routePlan
-predVeh : SimVehicle
-followVeh : SimVehicle
-speed : double
-position : double
-frictionModel : FrictionModel
-powertrainModel: PowertrainModel
-speedLimit : double
+simStep()
+simUpdate()
TrafficBlock
-enabled : bool
-greenGain : double
-status : int
-turnType: int
+isCompatibleFlow() : bool
+isEnabled () : bool
+simStep()
+simUpdate()

Figure 6.1: UML class diagram showing the relationship between the chosen components of the traffic simulator. The links starting with filled diamond shapes describe composition and can be read as 'is composed of'. Hollow diamond shapes represent aggregation, which is a weaker form of composition and can be read as 'has'. For example road is composed of lanes, one road can have many lanes but a lane can only belong to one road and a lane has vehicles on it but a vehicle can only be on one lane. Arrow shaped links are generalisations and can be read as 'is a type of'. For example an induction loop is a type of lane sensor.

tity based objects include the vehicles, traffic lights, intersections, sensors, wireless communication nodes. Roads and lanes are among passive components based on the *SimObject* class. The simulator simulates entities sequentially using a single processing thread.

In order to avoid situations where one entity wishes to interact with another but the other entity has not been processed yet, resulting in use of outdated data, the simulation cycle is split into two steps as shown in Figure 6.2. In the first step, the entity dynamic behaviour is simulated. In the case of a vehicle its physic model is used to simulate the response to current brake and throttle settings resulting in values such as position, speed and acceleration being calculated for the current simulation cycle. In the case of traffic lights, the phase of the signals has to be established before vehicles start reading it in the second phase. Once the dynamics

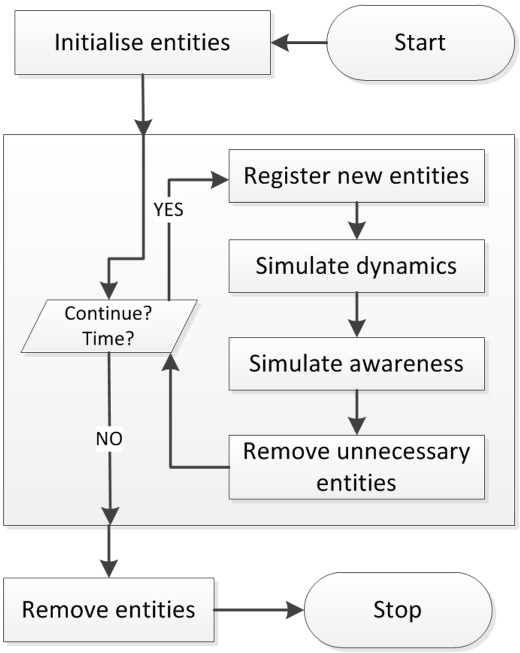

Initialise entities
Start
Register new entities
YES
Simulate dynamics
Continue?
Time?
Simulate awareness
NO
Remove unnecessary entities
Remove entities
Stop

Figure 6.2: The simulation cycle is divided into four steps. New entities are added to the simulation in the first step. The entity dynamics are simulated in the second step. The third step simulates situation awareness and unnecessary entities are removed from the simulation in stem four.

have been simulated for all entities it is safe to simulate the situation awareness. In this phase vehicles observe their surroundings, sensors detect vehicles and intersection control algorithms are run at previously determined sampling rates.

In order to ensure stability and determinability of the simulation the entity lists in *SimObjectRegistry* are locked while the simulation cycle executes. This means that entities cannot join or leave the simulation during a simulation cycle. However some entities can create other entities, e.g. *VehicleSpawner* creates vehicles. Those new entities are put on a waiting list and introduced to the simulation before the next simulation cycle starts. Removing entities from the simulation is achieved by flagging them as defunct during the simulation cycle. Such entities are then removed from the lists after the simulation cycle has been completed (see Figure 6.2).

6.3 Vehicle simulation

Vehicle behaviour simulation is one of the fundamental tasks of the simulator. Traffic flow is defined by the movement of vehicles and each vehicle in the simulation is a semi-independent agent that can have its behaviour customised. All vehicles share the same behaviour model but it can be parametrised to make each vehicle distinct.

Vehicles equipped with wireless communications are referred to as ITS-vehicles in this work. ITS-vehicles can communicate with the infrastructure using V2I enabling them to receive IATO advices (see Section 5.7 in Chapter 5) and dynamic routing information (see Section 3.6 in Chapter 3). They can also engage in cooperative platooning (see Subsection 6.3.6). Vehicles not equipped with wireless communication are referred to as normal or manual vehicles.

6.3.1 Vehicle dynamics - longitudinal behaviour

The vehicle simulation is considered semi-nanoscopic as it uses longitudinal dynamics models based on [136] together with simplified linear engine and brake models. This research project focuses on traffic management, therefore fully modelling all vehicle components, such as brakes, engine, powertrain and tyres was deemed unnecessary. The model was however designed with extensibility in mind to make it possible to introduce the aforementioned components in future.

Longitudinal vehicle dynamics are simulated in continuous linear space and discrete time. The basic longitudinal vehicle dynamics model can be expressed with a set of fundamental equations:

$$a_t = \frac{E(\alpha) - F(V_{t-1}) - B(\beta)}{M_v} \tag{6.1}$$

where α is the normalised (0%-100%) throttle setting and $E(\tau)$ is the modelled force output of the vehicle's powertrain. $F(V_{t-1})$ represents all friction associated with vehicles movement, including drag and tyre friction at time $t-1$. Braking force is represented by $B(\beta)$ with β being the normalised brake setting. In order to obtain the vehicle acceleration the sum of those forces is divided by the vehicle's mass M_v. The model extensibility is assured by realising this equation in software using polymorphic class hierarchy which will simplify adding different powertrain models as shown in Figure 6.3.

The acceleration is then used to calculate the distance the vehicle travelled d_t in during one time sample Δt such that:

$$d_t = V_{t-1}\Delta t + \frac{a_t(\Delta t)^2}{2} \tag{6.2}$$

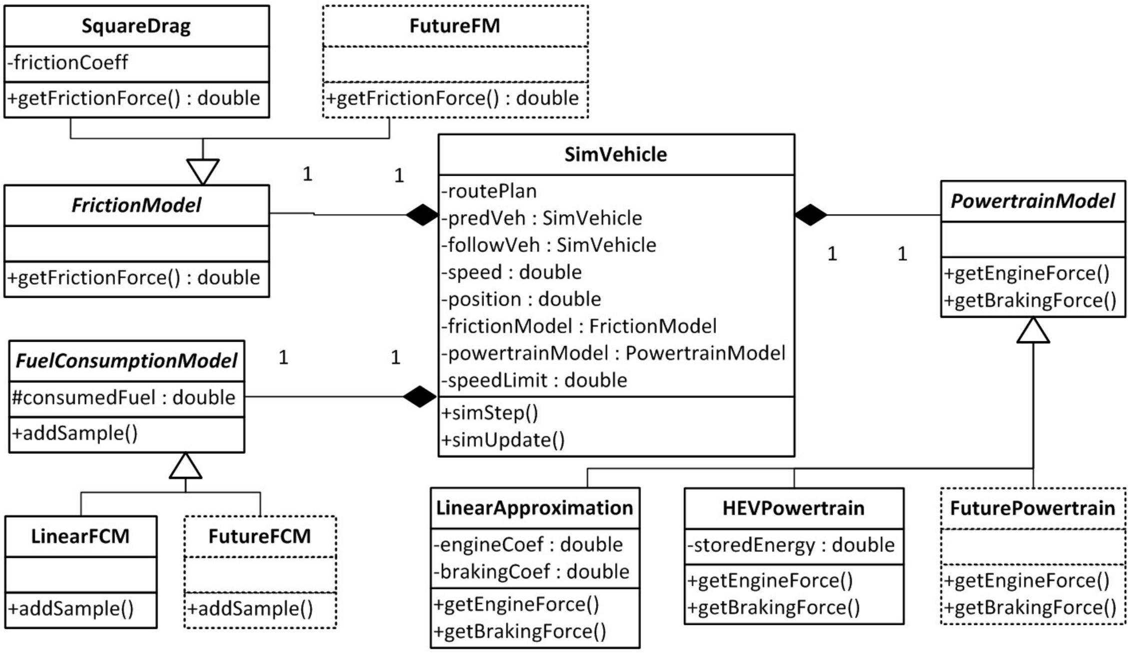

SquareDrag
-frictionCoeff
+getFrictionForce() : double
FutureFM
+getFrictionForce() : double
FrictionModel
+getFrictionForce() : double
SimVehicle
-routePlan
-predVeh : SimVehicle
-followVeh : SimVehicle
-speed : double
-position : double
-frictionModel : FrictionModel
-powertrainModel : PowertrainModel
-speedLimit : double
+simStep()
+simUpdate()
PowertrainModel
+getEngineForce()
+getBrakingForce()
FuelConsumptionModel
#consumedFuel : double
+addSample()
LinearFCM
+addSample()
FutureFCM
+addSample()
LinearApproximation
-engineCoef : double
-brakingCoef : double
+getEngineForce()
+getBrakingForce()
HEVPowertrain
-storedEnergy : double
+getEngineForce()
+getBrakingForce()
FuturePowertrain
+getEngineForce()
+getBrakingForce()
1
1
1
1
1
1

Figure 6.3: Class diagram of a simulated vehicle and its components.

where the vehicle speed V_t at time t is given by:

$$V_t = \frac{d_t}{\Delta t} \tag{6.3}$$

Once the vehicle behaviour has been simulated its position p_t relative to the beginning of the lane is updated as follows:

$$p_t = p_{t-1} + d_t \tag{6.4}$$

Vehicle dynamics - longitudinal model components

In order to keep the vehicle model simple a linear relationship was assumed between engine force $E(\alpha)$ and throttle setting α as well as between braking force $B(\beta)$ brake cylinder pressure β. Therefore $E(\alpha)$ from Equation 6.1 can be expressed as:

$$E(\alpha) = \alpha C_E \tag{6.5}$$

where C_E is the engine force coefficient and α is the throttle setting. Similarly the braking function has been approximated by:

$$B(\beta) = \beta C_B \tag{6.6}$$

where C_B is the braking force coefficient and β is the brake setting (pressure in the brake cylinder).

The final component of the model aims to represent all the friction effects that occur in a moving vehicle, such as air drag, roll friction and powertrain resistance. It is modelled using a second order polynomial:

$$F(V_t) = C_{F2}V_t^2 + C_{F1}V_t + C_{F0} \tag{6.7}$$

where $C_{F\{0,1,2\}}$ are the tunable friction coefficients and V is the current vehicle speed.

The powertrain losses would have to be described using a parametric function, depending on the gear configuration, however since the powertrain is not taken into account in this model, powertrain related losses are assumed to be incorporated in the friction modelling component.

The energy consumption model is based on a simplifying assumption that the amount of consumed energy is proportional to the forward force generated by the vehicle propulsion unit. It has been constructed based on the fuel consumption modelling function in [137]. The engine is approximated using Equation 6.5 and assumes that the forward force is proportional to the throttle setting. Therefore the energy consumption e_t in time sample t can be approximated by:

$$e_t = \tau C_E \Delta t \tag{6.8}$$

Where C_E is the engine force coefficient and Δt is the sample length.

In order to approximate real vehicle behaviour the model was tuned by selecting appropriate values for the tuning parameters C_E, C_B and $C_{F\{0,1,2\}}$.

6.3.2 Vehicle dynamics - longitudinal model tuning

The previous section described the longitudinal vehicle model used by the traffic simulator. This section describes the tuning process of that model.

The model input consists of the throttle setting and the brake cylinder pressure. The measured outputs consisted of the vehicle speed and acceleration (see Figure 6.4).

Measurement data used to tune and validate the model was obtained using the Network Assisted Vehicle (Ford Mondeo) vehicle described in [138] (see Appendices F and G). The measurements were carried out on a straight, flat road to minimise the impacts of road gradient and dynamics changes resulting from the vehicle performing lateral manoeuvres.

An iterative improvement process was used to tune the vehicle model. Such an approach explored the parameter range with the aim to minimise the mean square error (MSE) between the modelled and measured acceleration. The modelled acceleration samples are given by vector $\hat{A}$ and the measured acceleration values are

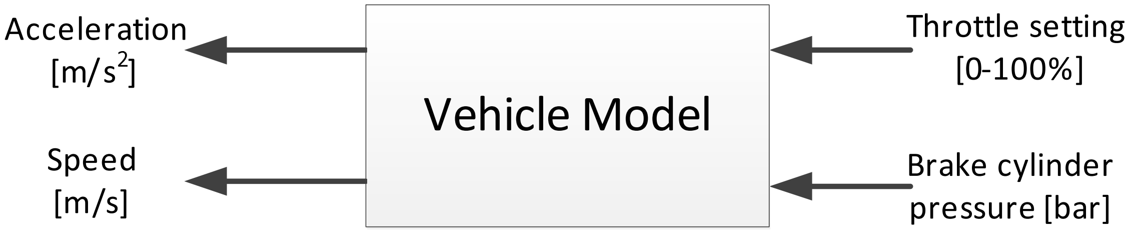


Figure 6.4: The vehicle model used in the simulator. Throttle and brake settings form the model input and speed and acceleration are the model outputs.

represented by the vector A. The MSE is given as follows:

$$MSE(\hat{A}) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{A}_i - A_i\right)^2 \tag{6.9}$$

Using a good initial value of the tuning parameters the performance of the iterative improvement algorithm can be increased. In order to take advantage of that fact and to obtain as good parameters as possible the tuning process was carried out in two phases. In the first phase the parameters of each subcomponent were tuned semi-independently by minimising or eliminating the impact of the remaining subcomponents. In the second phase the model was fine tuned using data obtained by performing a comprehensive drive cycle with various amount of braking and accelerations. The first phase consisted of:

1. Step 1 - Obtain friction component coefficients
   Friction is present and affects the overall model, therefore the friction coefficients $C_{F\{0,1,2\}}$ were obtained first based on measurements obtained from the vehicle coasting down from $90km/h$ to $10km/h$.
2. Step 2 - Tune the engine modelling component
   The engine component coefficient C_E is tuned using a data set obtained by making the vehicle perform various accelerations. Using the initial approximation of the friction component parameters the same process as in Step 1 is used with a dataset comprising a number of repeating sequences of various magnitudes of accelerations. Brakes are not used in this scenario and the vehicle is allowed to coast down before the following acceleration is performed.
3. Step 3 - Tune the brake modelling component
   The set-up in this case is similar to the previous case, however various amounts of brake pressure are applied to slow the vehicle down instead of allowing it to coast down.

Using the initial parameter estimates, the model was then fine tuned using a data set containing a comprehensive set of longitudinal vehicle behaviours. Figures 6.5 and 6.6 illustrate the measured and modelled vehicle speeds and accelerations in different drive cycles. It is visible that despite using a simplified model (see Subsection 6.3.1) the acceleration is modelled with satisfying accuracy. The speed errors are cumulative due to the model being simulated in open loop and result in the speed mismatch observed in Figure 6.6.

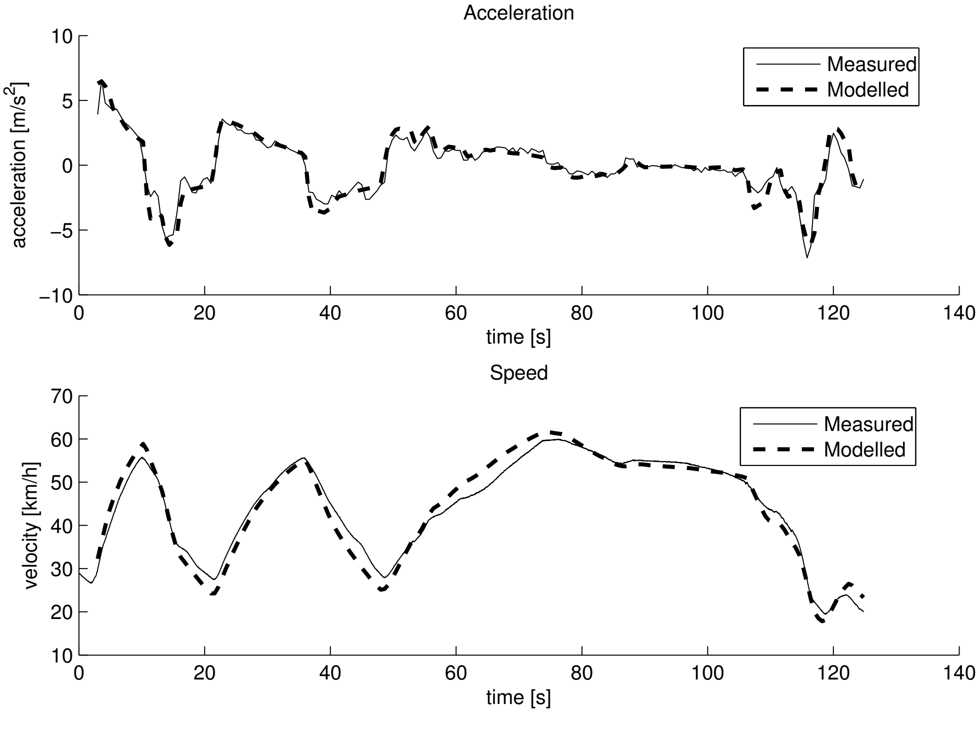


Figure 6.5: Modelled and measured vehicle speed and acceleration - tuning data set

Table 6.1: Model tuning results

	Tuning data set		Validation data set	
	MSE	RMSE	MSE	RMSE
speed	3.55 $\left[\frac{m}{s}\right]^2$	1.88 $\left[\frac{m}{s}\right]$	5.41 $\left[\frac{m}{s}\right]^2$	2.32 $\left[\frac{m}{s}\right]^2$
acceleration	0.3 $\left[\frac{m}{s^2}\right]^2$	0.55 $\left[\frac{m}{s^2}\right]$	0.22 $\left[\frac{m}{s^2}\right]^2$	0.47 $\left[\frac{m}{s^2}\right]$

MSE (see Equation 6.9) and root MSE (RMSE) were used to evaluate the accuracy of the vehicle model, see Table 6.1. The vehicle model parameters obtained in the tuning process are presented in Table 6.2.

Table 6.2: Mean Square Errors (MSE)

parameter	value
C_E	456.3031
C_B	577.4780
C_{F0}	1243.1
C_{F1}	5.4983
C_{F2}	1.3427

In order to create a realistic simulation environment with different vehicle types the vehicle model parameters C_E, C_B, C_{F0}, C_{F1} and C_{F2} are randomised ±10% for each vehicle injected into the simulation. Figure 6.7 illustrates the reaction of the vehicle model to the same input under different sets of tuning parameters.

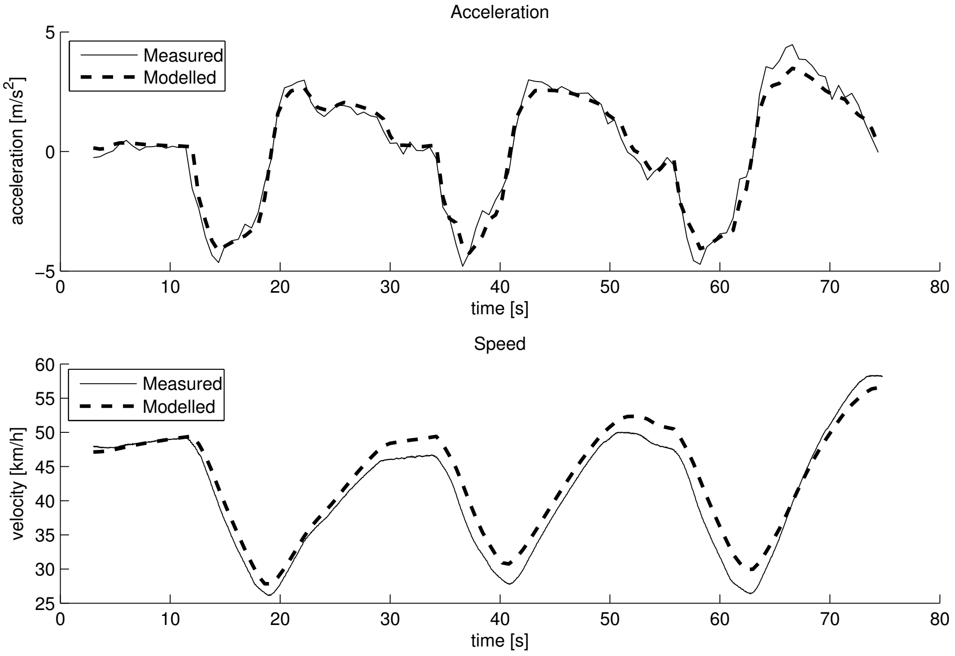


Figure 6.6: Modelled and measured vehicle speed and acceleration - validation data set

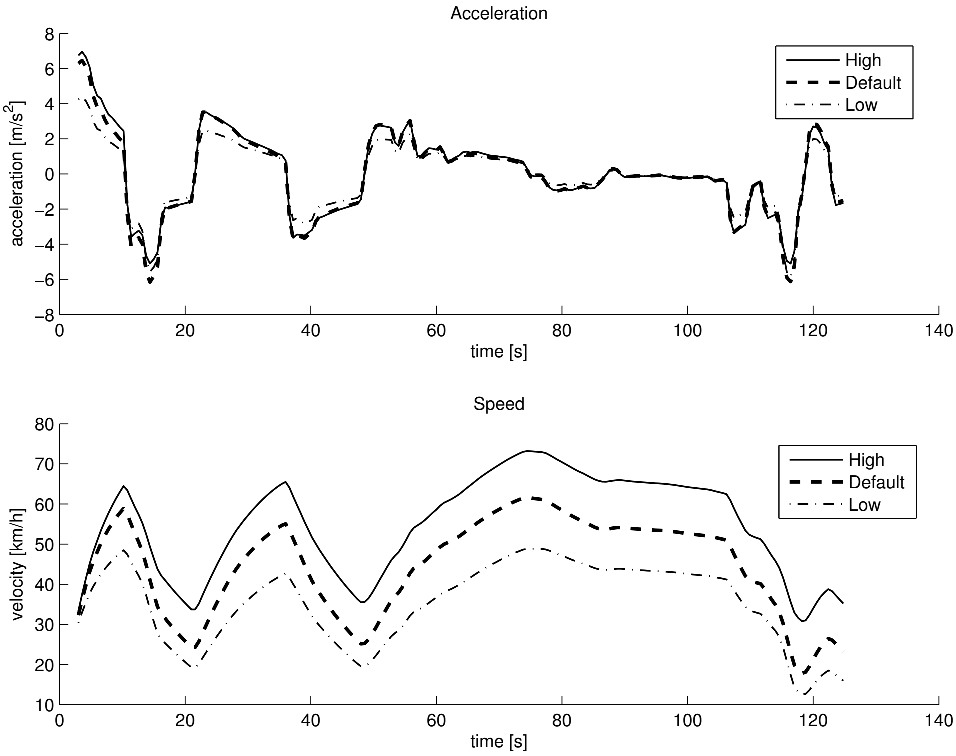


Figure 6.7: The reaction of vehicle model to extreme ranges of parameters. High extreme, low extreme and average is presented.

The default variant uses the parameter set presented in Table 6.2. The high extreme variant represents a vehicle which compared to the default variant has proportionally higher engine power to the resistance and braking forces. The high extreme variant is obtained by setting the tuning parameters such that $C_E^{high} = 1.1C_E$, $C_B^{high} = 0.9C_B$, $C_{F\{0,1,2\}}^{high} = 0.9C_{F\{0,1,2\}}$. The low extreme vehicle model represents situation opposite to what was described above and uses the following set of parameters $C_E^{low} = 0.9C_E$, $C_B^{low} = 1.1C_B$, $C_{F\{0,1,2\}}^{low} = 1.1C_{F\{0,1,2\}}$.

6.3.3 Vehicle speed control

In this simulator there are two controllers managing lateral vehicle behaviour. The high level controller generates a speed demand based on the current situation on the road. Its goal is to make the vehicle travel at the maximum allowed speed, or to maintain an appropriate separation from the preceding vehicle. The low level controller actuates the throttle and brakes of the vehicle (see Subsection 6.3.1) to achieve the speed set by the high level controller.

The low level controller comprises a pair of PID controllers with appropriate switching logic to prevent both throttle and brake usage at the same time. Switching between throttle and brakes is not instantaneous. The brakes to throttle switching time $T_{brk,thr}$ is 150 milliseconds whereas the change from throttle to brakes is faster and set to $T_{thr,brk} = 100$ milliseconds Vales of $T_{thr,brk}$ and $T_{brk,thr}$ were chosen based on the switching times observed in a real vehicle (see Appendix G). A *Simulink* equivalent to the low level vehicle controller can be seen in Figure 6.8.

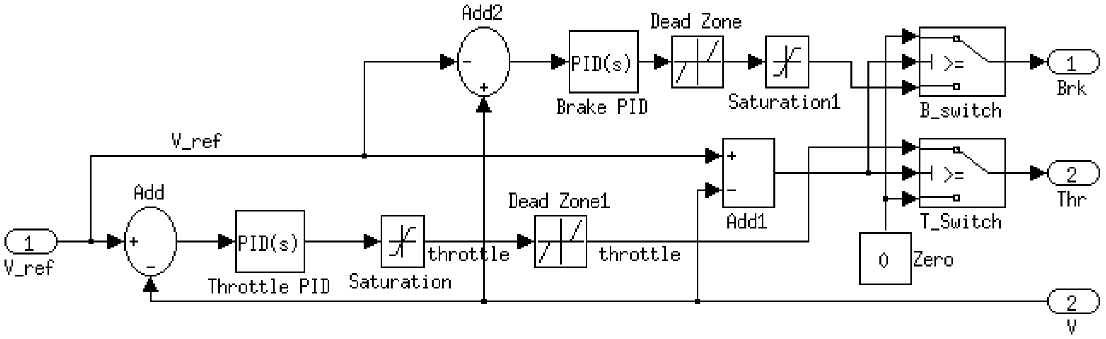

Add2
Dead Zone
PID(s)
Brake PID
Saturation1
>=
B_switch
1
Brk
V_ref
Add1
>=
T_Switch
2
Thr
Add
Dead Zone1
1
V_ref
PID(s)
Throttle PID
Saturation
throttle
throttle
0
Zero
2
V

Figure 6.8: The Simulink equivalent of the low level vehicle controller implemented in the traffic simulator. It comprises two PID controllers governing the brake and throttle setting (see Subsection 6.3.1 and Figure 6.4). Switching logic prevents simultaneous use of throttle and brake usage.

6.3.4 Lane changing - lateral behaviour

The previous sections described the creation, tuning and control of the longitudinal vehicle model. This section describes the lateral component of vehicle modelling.

Lane changing is a common occurrence in any traffic conditions and has a great impact on traffic flow [139, 140]. There are two main reasons for the vehicle to change lanes. The change might be routing related, when a vehicle is approaching an intersection and the lane it occupies does not lead to its desired destination. A vehicle may also decide to change lane on its own, when it cannot go as fast as it desires on the lane it currently occupies. This decision making process is inspired by human reasoning in such situations. The vehicle will want to change lanes when it is, according to a cost function (see Equation 6.11), worth doing [102, 141]. A lane change usually occurs when approaching or following another vehicle which is moving slower than the current vehicle wishes to go. A lane change desire will be indicated only if it will improve the locally perceived situation of the vehicle. For example if there are slowly moving vehicles on all lanes, a lane change will not be beneficial and therefore will not be carried out.

The simulator handles lane changing as a three-step process as shown in Figure 6.9. The first step determines whether the vehicle is satisfied with the conditions on the lane it is currently occupying (see Equation 6.10), the second step is to generate the numerical value of lane change desire or requirement based on the conditions on the current and prospective lanes (see Equation 6.11). The third and final step is to check if it is safe to proceed. The first and second steps are applied only to traffic situation induced lane changes. The process of performing a routing related lane change starts at step three.

The perceived lane satisfaction level is deemed unsatisfying if the following inequality is satisfied:

$$\frac{V_p}{V_{desired}} < \iota_i \tag{6.10}$$

If the preceding vehicle's speed V_p relative to the current vehicle's desired speed $V_{desired}$ is below the current vehicle's lane change threshold value ι_i the vehicle will attempt to change lanes. If there is no preceding vehicle present, the traffic conditions induced lane change is not necessary.

The second step investigates the situation on other lanes and determines if performing a lane change would be beneficial. This is modelled using cost function based algorithm where the cost of changing lanes is calculated with respect to the

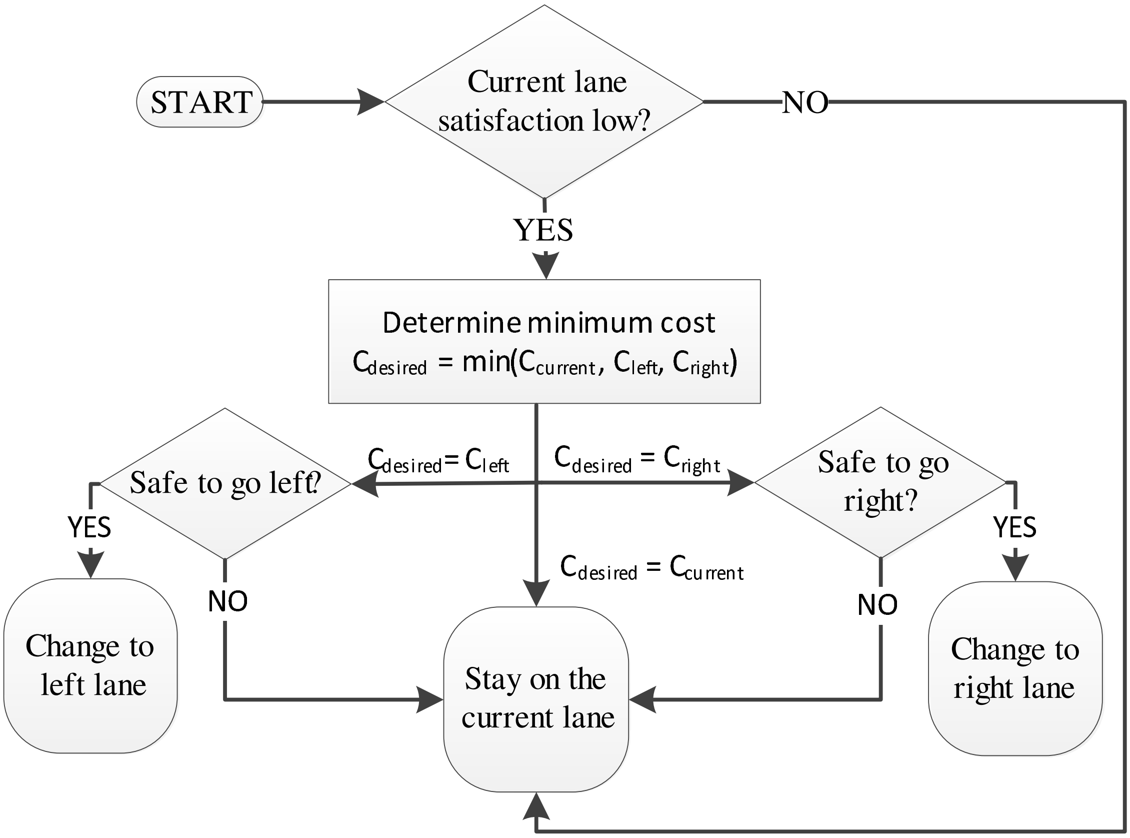

START
Current lane satisfaction low?
NO
YES
Determine minimum cost
$C_{desired} = \min(C_{current}, C_{left}, C_{right})$
Safe to go left?
$C_{desired} = C_{left}$
$C_{desired} = C_{right}$
Safe to go right?
YES
NO
$C_{desired} = C_{current}$
NO
YES
Change to left lane
Stay on the current lane
Change to right lane

Figure 6.9: Lane change decision diagram. A vehicle will change lanes when it is dissatisfied with its current lane (see Equation 6.10), there is a better lane visible (see Equation 6.11) and it is safe to perform the manoeuvre (see Equations 6.12 and 6.14).

potential preceding vehicle:

$$C(s_p, \Delta V_p) = \begin{cases} \frac{\alpha_s}{s_p} + \alpha_v \Delta V_p, & s_p < s_{max|p} \\ 0, & s_p \geq s_{max|p} \end{cases} \tag{6.11}$$

Where s_p represents the distance between the current vehicle and the preceding vehicle should the lane change occur. Similarly ΔV_p reflects vehicles' relative speed, α_s , α_v are tunable weights that are part of the driver behaviour modelling (see Subsection 6.3.8) and $s_{max|p}$ represents the preceding vehicle detection range.

The cost function is calculated for the current lane and both immediate neighbouring lanes, if they exist. The lane with the lowest cost is chosen and if it is different to the currently occupied lane, the lane change desire is indicated.

Both desired and required (topological) lane changes are subject to safety checks which determine if it safe to proceed. The checks are carried out with respect to the separation gap and relative speed, between the current and the potential preceding and following vehicles. The safe separation for lane changing is determined using the same time headway policy that controls the separation between vehicles in a platoon (see Subsection 2.4.1 of Chapter 2). It is assumed that the separation is sufficient to change lanes provided the following inequalities are satisfied:

$$\begin{aligned} s_p > &\quad s_{0|p} + V \beta_{s|p} \\ s_f > &\quad s_{0|f} + \tilde{V}_f \beta_{s|f} \end{aligned} \tag{6.12}$$

Where s_p and s_f denote the separation distances from the potential preceding (p) and following (f) vehicles, the speed of the current vehicle is given by V and $\tilde{V}_f$ represents the speed of the potential following vehicle. Parameters $s_{0|p}$, $s_{0|f}$ represent the minimal constant separation and $\beta_{s|p}$ and $\beta_{s|f}$ correspond to the time headway spacing policy. $\beta_{s|p}$ and $\beta_{s|f}$ can vary with the driver type (see Subsection 6.3.8) but their default values are given as:

$$\begin{aligned} s_{0|p} = & \quad s_{0|f} = d_0 \\ \beta_{s|p} = & \quad \beta_{s|f} = \tau \end{aligned} \tag{6.13}$$

where d_0 and τ are the time headway policy parameters described in Subsection 6.3.5.

Assuring sufficient separation is a necessary condition but it is not sufficient to ensure safe lane changing. The second set of checks investigates the speed differences between the current vehicle and its potential follower and predecessor:

$$\begin{aligned} \Delta\tilde{V}_p < & \quad s_p\beta_{v|p} \\ \Delta\tilde{V}_f > & \quad -s_f\beta_{v|f} \end{aligned} \tag{6.14}$$

Those checks ensure that the higher the speed difference is, the longer the separation is to safely change lanes. Note that those checks only apply if the potential follower is travelling faster than the current vehicle and the potential predecessor is moving slower. The relative speeds between the current vehicle and a potential predecessor $\Delta\tilde{V}_p$, and the potential follower $\tilde{V}_f$ are defined as:

$$\begin{aligned} \Delta\tilde{V}_p = & \quad V - V_p \\ \Delta\tilde{V}_f = & \quad V - V_f \end{aligned} \tag{6.15}$$

where V is the speed of the vehicle that is considering changing lanes (see Figure 6.10)

If the lane change desire has been expressed, but the safety constraints have not been satisfied the desire indication is cleared, but will be re-assessed in the next simulation cycle to take into account changes in traffic conditions. If all of the safety requirements have been fulfilled the simulator relocates the vehicle on to the desired lane.

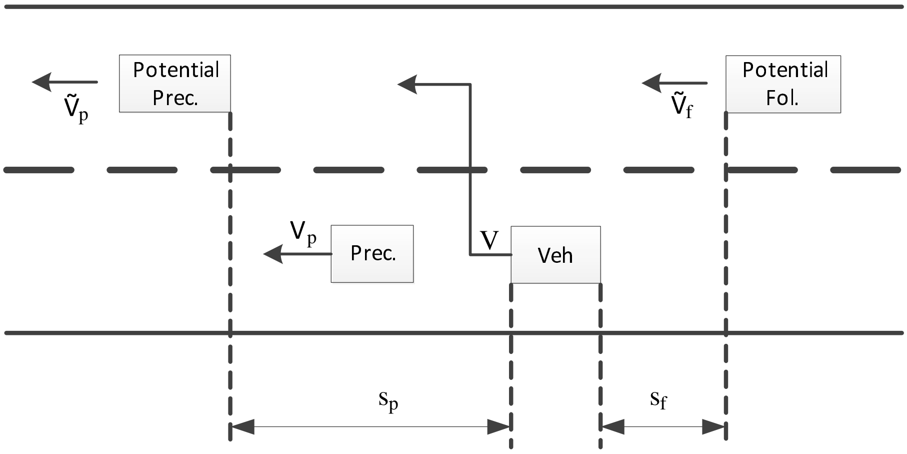


Figure 6.10: Illustration of the lane changing mechanism.

6.3.5 Vehicle following (Platooning)

If any given vehicle encounters a slower moving vehicle and cannot overtake it by performing a lane change it has no choice but to follow. If two or more vehicles engage in such behaviour a platoon is formed. It is assumed that each vehicle in a platoon, apart from the platoon leader, desires to travel faster than its predecessor (otherwise the platoon would not be formed or it would break up into smaller platoons), however it is prohibited from doing so by the vehicle in front of it. In such a case the vehicle can only travel as fast as its predecessor while keeping an appropriate distance.

The low level vehicle controller (see Subsection 6.3.3) maintains desired speed by actuating brakes and throttle. It is the task of the high level controller to generate such speed demand. If the road in front of the vehicle is empty the task of the high level controller is reduced to setting the speed limit as the desired speed. The situation is different when the vehicle has to follow another. In such a case it aims to move with the same speed as its predecessor while maintaining an appropriate separation gap. If the gap is too small/large the speed demand should be decreased/increased to maintain the required gap. The desired speed difference is given as follows:

$$\Delta V_{desired} = \Delta s_p \ \beta_{d|ACC} \tag{6.16}$$

Where $\beta_{d|ACC}$ is an empirically chosen tuning parameter and Δs_p is a separation error given by:

$$\Delta s_p = s_p - s_p^{desired} \tag{6.17}$$

Where s_p is the current separation between the vehicles. Desired separation $s_p^{desired}$ is obtained using constant time headway spacing policy [15], defined as follows:

$$s_p^{desired} = s_0 + V\tau \tag{6.18}$$

Where s_0 is a minimal constant separation (set to s_0=1 meter to make the vehicles maintain spacing even if stopped), V is the vehicle speed and τ is the time headway.

The simulator uses an experimentally established constant time headway of τ=1 second (see Subsection 7.8.3 of Chapter 7), however this might be revised to variable time headway policy in nearby future as this area of vehicle control holds a great promise in increasing traffic performance [90].

Once the desired speed difference $\Delta V_{desired}$ is obtained the speed demand can be calculated as follows:

$$V_i = \begin{cases} min(V_{max},\ V_p + \Delta V_{desired}), & \Delta s < s_{p,max} \\ V_{max}, & \Delta s \geq s_{p,max} \end{cases} \tag{6.19}$$

Where $s_{p,max}$ is the preceding vehicle detection distance that depends on the road speed limit V_{max}.

$$s_{p,max} = V_{max}\beta_{sMax} \tag{6.20}$$

β_{sMax} defines how long it takes for a vehicle to close the distance to the detected object while travelling at V_{max}. Setting β_{sMax} to 5 seconds has shown to be sufficient for all simulated scenarios.

Having obtained current speed demand V_i, the low level controller is used to determine throttle and brake settings for the next simulation cycle (see Subsection 6.3.3).

6.3.6 Cooperative platooning

When performing standard platooning a vehicle can reliably take into account only the vehicle in front of it. In some instance, it can observe further into the platoon but such visibility cannot be guaranteed. This can lead to inefficient vehicle following behaviour.

Cooperative platooning or CACC aims to increase the platoon performance by providing vehicles with advance information about platoon leader's current behaviour and intent using wireless V2V communication, see Section 6.4. This information makes it possible to coordinate vehicle manoeuvres increasing the stability of the platoon [41, 93, 96] and allowing smaller inter-vehicle gaps [41] leading to greater road utilisation and therefore increased overall traffic performance.

Every vehicle in a platoon is aware at least of the platoon leader's speed and in some occasions can access other data such as intended acceleration or deceleration [96]. In this work, all vehicles in a CACC platoon have access to the platoon leader's current speed and current target speed which the lead vehicle is trying to achieve.

The platoon leader uses ACC to govern its speed and the speed demand for the followers is generated using CACC such that:

$$V_i = \begin{cases} V_{lead}^{desired} + \Delta V_p \beta_{v|CACC} + \Delta s_p \beta_{d|CACC}, & \Delta s_p < s_{max} \\ V_{max}, & \Delta s_p \geq s_{max} \end{cases} \tag{6.21}$$

Note that this equation is similar to (6.19) but instead of using the preceding vehicle speed (V_p) it uses the platoon leader's desired speed $V_{lead}^{desired}$ as the base speed demand. Even though the preceding vehicle speed V_p is not the base for the speed demand calculation the vehicle will still aim to maintain an appropriate separation form its predecessor and keep their relative speed ΔV_p low. s_p is calculated the same way as in ACC platooning approach described in the previous section (see Equations 6.17 and 6.18), however due to superior string stability of CACC platooning over ACC (see Subsection 2.4.1 of Chapter 2) it is possible to use smaller time headway with CACC platoons. In this work $\tau_{CACC} = 0.6$ seconds was used. The expression $\Delta V_p \beta_{v|CACC}$ aims to minimise the speed differences between the consecutive vehicles.

The parameters $\beta_{v|CACC}$ and $\beta_{d|CACC}$ are used to tune the CACC platooning controller. Increasing their value will reduce the influence of the platoon leader and make the vehicle prioritise the minimisation of the speed difference to maintain the separation gap from the immediate predecessor constant. Reducing those values will have the opposite effect and increase the influence of the platoon leader at the cost of the separation gap. The values of $\beta_{v|CACC} = 1.9$ and $\beta_{d|CACC} = 3.5$ were used.

6.3.7 Off-board speed advice

The ITS-vehicles are capable of wireless communication and can benefit from receiving off-board speed advice from the traffic management system. The simulator supports two types of off board speed requests. The first type is an explicit speed limit that replaces the current lane speed limit V_{max} (see Euations 6.19 and 6.21). The second type is the intersection approach advice issued by the Intersection Approach Trajectory Optimisation (IATO). In such case the vehicle is provided with the time until the traffic light changes, the direction of change (red to green or green to red) and the location of the stop line. Knowing its position, the vehicle can approximate the distance to the stop line and calculate the approach trajectory as described in Section 3.4 of Chapter 3. The vehicle follows the calculated trajectory unless the road situation requires further deceleration.

The difference between the explicit speed limit and IATO speed advice is that if the vehicle receives a new speed limit, usually by entering a new speed limit zone or passing a speed limit sign, it will brake to slow down if necessary. If the vehicle travels faster than the speed advice dictates it will coast down to appropriate speed without using brakes.

6.3.8 Driver modelling

The driver is a crucial component of the vehicle's control system that performs the high level control over the vehicle behaviour. Its task is to safely guide the vehicle to its intended destination while obeying various road rules, interacting with other road users and handling unforeseen situations. Work of several researchers has been summarised in [142] claiming that due to very complex feedback system between the driver and the vehicle, the driver-vehicle system has to be treated as a whole. Such approach was adopted when designing the traffic simulator and the driver model was constructed as an integral part of the vehicle model. The tasks performed by the human driver have been identified as:

- path following

  The driver is responsible for getting the vehicle to the intended destination. This includes changing lanes when appropriate and taking appropriate turns at intersections.
- obstacle avoidance

  The driver has to react to frequently changing traffic conditions to avoid collisions with other traffic participants and other objects that might be on road.
- headway control

> It is the driver's responsibility to maintain an appropriate separation gap from preceding vehicle. The size of the separation gap is a trade-off between road utilisation and safety of traffic participants.

In the traffic simulator tool the path following problem is reduced to the selection of the most appropriate lane. It is due to the method chosen to simulate lateral vehicle motion (see Subsection 6.3.1). Obstacle avoidance as such is not necessary as the simulator does not support simulating non-vehicular objects on lanes. Headway control is simulated in detail, as described in Subsection 6.3.5.

The driver model is therefore defined by the ACC headway control and the lane changing mechanism, both of which can be tuned to reflect different driver personalities. The CACC platooning is based on automated vehicle control and therefore is not affected by human driver modelling. Drivers that tend to accept lower safety margin when changing lanes, maintain smaller gap from the vehicle in front and are more likely to overtake have been termed aggressive drivers. On the other hand drivers that tend to avoid overtaking, always leave bigger safety margins when changing lanes and maintain large separation gaps are referred to as prudent driver in this work.

In order to reflect the driver personality the parameters governing the decision process of both the longitudinal and lateral vehicle control components are modified accordingly. Increasing the lane dissatisfaction threshold parameter ι_i (see Equation 6.10) will cause the driver to be dissatisfied with the occupied lane more often resulting in much more frequent desire to change lanes. Decreasing the lane changing safety parameters $\beta_{s|p}$ and $\beta_{s|f}$ (see Equation 6.12) as well as increasing $\beta_{v|p}$ and $\beta_{v|f}$ (see Equation 6.14) will result in the driver to accept lower safety margins when performing lane changes.

The parameters mentioned above have been chosen experimentally to make the vehicles compromise between maximising the benefit from choosing a best lane and minimising the negative impact on other vehicles resulting from doing so. Those values of the parameters correspond to the normal driver behaviour in the context of this work.

In the headway control mechanism the driver personality affects the time headway τ (see Equation 6.12), which determines the separation gap the vehicle aims to maintain. Aggressive drivers will have the time headway lowered which will result in decreased separation gaps and sharper accelerations and decelerations. Prudent drivers will use an increased time headway and therefore maintain higher separation gap.

The driver's personality is determined by the driver aggravation coefficient ξ which adjusts the model parameters as follows:

$$\begin{aligned} \bar{\iota_i} &= \iota_i(1+\xi) \\ \bar{\beta_{v|p}} &= \beta_{v|p}(1+\xi) \\ \bar{\beta_{v|f}} &= \beta_{v|f}(1+\xi) \\ \bar{\beta_{s|p}} &= \beta_{s|p}(1-\xi) \\ \bar{\beta_{s|f}} &= \beta_{s|f}(1-\xi) \\ \bar{\tau} &= \tau(1-\xi) \end{aligned} \tag{6.22}$$

where $\xi \in [-0.1, 0.1]$ where the lowest value corresponds to prudent driver, and the highest value results in aggressive driver behaviour. The parameter ξ is chosen randomly from the specified range, using uniform distribution, for every new vehicle joining the simulation provided that the appropriate option is selected.

6.4 Simulation of wireless communication

Wireless Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications are among the fundamental technologies that enable ITS. In this work V2V communications are used to enable cooperative platooning (see Subsection 6.3.6) and V2I communications enable IATO speed advice (see Subsection 6.3.7.

One of the technologies enabling such communication is IEEE 802.11p based DSRC [113] system discussed in Section 2.5 of Chapter 2. Extensive studies on the performance of IEEE 802.11p based DSRC systems have been carried out in [143], where the authors investigated the performance of that technology in various scenarios and conditions. The performance of DSRC was observed to be very variable and heavily dependent on the environment, however on average the system performed well achieving high packed delivery ratios.

Independent studies of 802.11p based DSRC systems have been carried out at MIRA as part of the NAV project [138]. Point to point communications were tested on MIRA proving ground, using three different 802.11p hardware devices, and measures such as latency and packet delivery ratio were examined with respect to the type of hardware used, separation distance and relative velocity between the nodes.

Figure 6.11 shows the average delay incurred in one of the examined hardware with respect to distance between the communicating node. Even though the communication was possible at large distances, due to the application of DSRC only the first 1000 meters were taken into account. It is visible that that latency within the selected range was varying between 18 ms and 22 ms. The obtained results are in line with [97] where 20 milliseconds network latency was assumed by the authors.

The simulator does not possess dedicated network simulation capability at this point and the wireless communication is simulated in the form of transmission latency between the sender and the receiver. Due to the discrete sampling space used in the simulation the minimal simulated latency is one time sample. The sampling time of 100 milliseconds used in the simulator was sufficient to account for both the wireless communication latency and processing delays. Implementation of extended network simulation or integration with an external network emulation tool [111] is considered for future work.

6.5 Road network simulation

The simulated traffic network is composed of roads, lanes and intersections. It is defined as a directed graph where intersections are vertexes and roads are the edges.

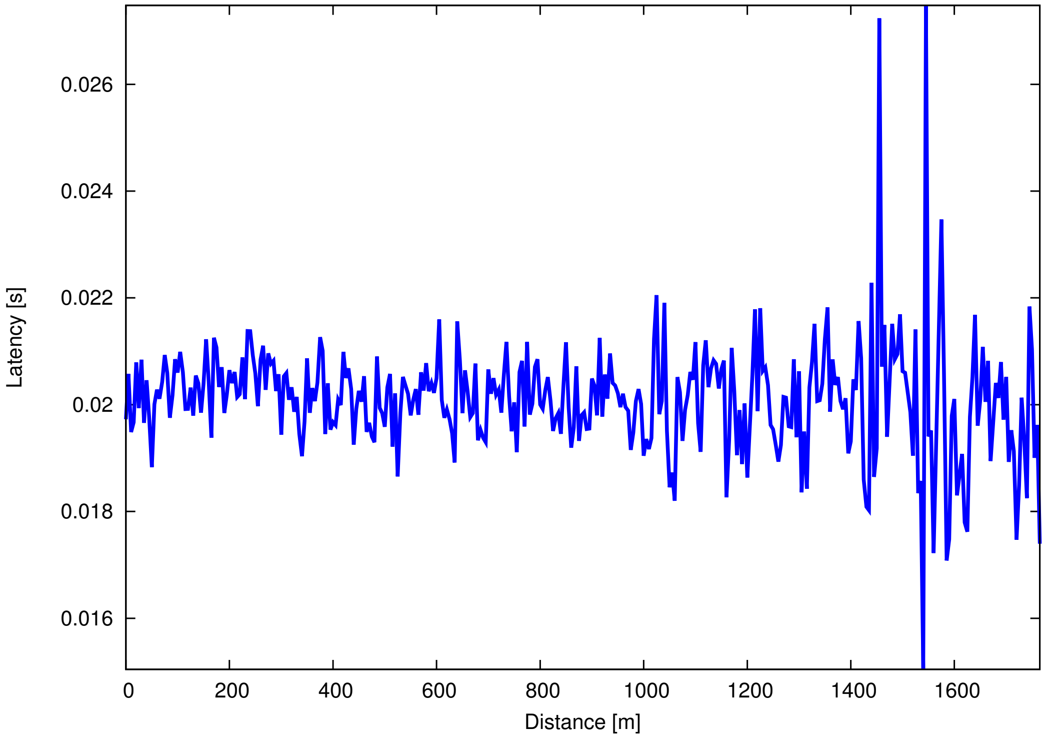


Figure 6.11: DSRC latency with respect to separation distance measured on MIRA proving ground. Low latency between 18 and 22 milliseconds observed in range of 1000m. Increased latency observed beyond that range due to retransmissions triggered by packet loss.

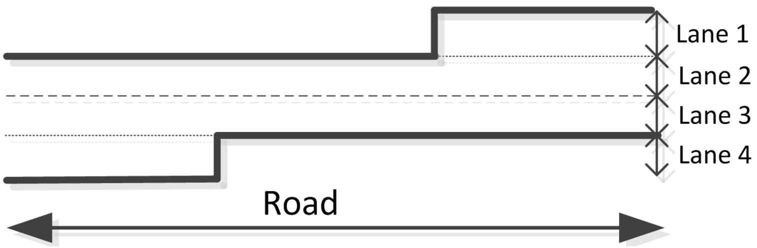


Figure 6.12: Example lane configuration. The simulator allows creation of lanes that span a fraction of the road length. This allows ramps or dedicated intersection turn lanes to be created.

6.5.1 Road simulation

A road is defined as a bidirectional link between intersections with a specified numbers of lanes in each direction and a defined top speed limit, which can be overridden by placing appropriate static signs or VMS. Lanes can span the entire length of the road or just its sections. Such lane configuration makes it possible to define turning lanes on the intersections or ramps. Figure 6.12 illustrates an example of a road comprising unidirectional lanes, two of them span the entire length of the road and the remaining two are available just in its sections.

The lanes belonging to each direction of a road hold references to adjacent lanes to enable vehicles to change lanes. A one way street can be defined by setting the amount of lanes in one of the directions to zero.

6.5.2 Intersection simulation

An intersection is a point where two or more roads converge. It allows the vehicles to advance from one road link to another. Lanes from different roads are joined and the resulting flow compatibility is defined and regulated by the intersection object.

Each intersection has connection nodes to which the roads can be attached. Depending on the intersection type there can be several ways of joining roads in an intersection. The simulator has a basic cross type of intersection implemented

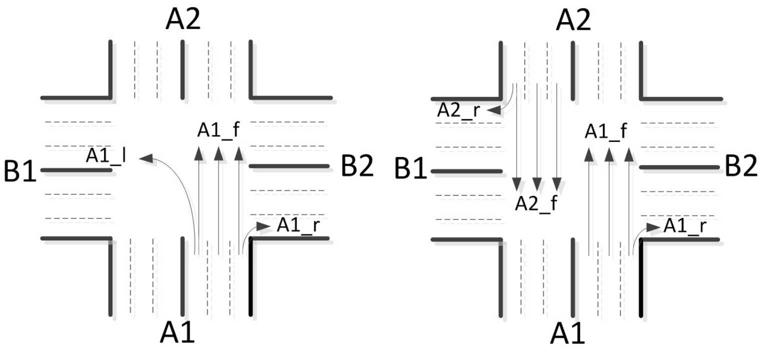


Figure 6.13: Cross type intersection flow compatibility for right hand traffic. Nodes A1, A2, B1 and B2 are used by the simulator to attach roads to the intersection.

and an interface is provided that can be used to define custom intersection types.

The intersection definition holds traffic flow compatibility rules. In the simplest cross-type intersection each node, marked A1, A2, B1, B2, can have three outflow destinations indicated by the subscript L, F and R for left forward and right respectively, see Figure 6.13.

The simulator automatically generates signalling phases based on flow compatibility

Table 6.3: Intersection flow compatibility truth table. Value of 1 indicates compatibility between the flows and lack of thereof is indicated by 0.

		A1			**A2**			**B1**			**B2**		
		L	**F**	**R**	**L**	**F**	**R**	**L**	**F**	**R**	**L**	**F**	**R**
A1	**L**	1	1	1	1	0	0	0	0	1	0	0	1
	F	1	1	1	0	1	1	0	0	1	0	0	0
	R	1	1	1	0	1	1	1	0	1	1	1	1
A2	**L**	1	0	0	1	1	1	0	0	1	0	0	1
	F	0	1	1	1	1	1	0	0	0	0	0	1
	R	0	1	1	1	1	1	1	1	1	1	0	1
B1	**L**	0	0	1	0	0	1	1	1	1	1	0	0
	F	0	0	0	0	0	1	1	1	1	0	1	1
	R	1	1	1	1	0	1	1	1	1	0	1	1
B2	**L**	0	0	1	0	0	1	1	0	0	1	1	1
	F	0	0	1	0	0	0	0	1	1	1	1	1
	R	1	0	1	1	1	1	0	1	1	1	1	1

```
Procedure GenerateStages():
stageList = empty list of stages
for each flowA from Intersection do
    stage = empty list of compatible flows
    add flowA to stage
    for each flowB from Intersection do
        if flowA compatible with flowB then
            add flowB to stage
        end
    end
    if stage does not exist in stageList then
        add stage to stageList
    end
end
```

Algorithm 6: Dynamic signalling stage generation

matrices that describe the compatibility between the different ways of traversing the intersection. Table 6.3 provides the flow compatibility information for a cross-type intersection for right hand traffic (see Figure 6.13). The simulator has a similar table for use with left hand traffic. Such table can be utilised to generate a list of all possible combinations of signalling stages, where a stage is composed of compatible flow directions that can be given green phase simultaneously. It is achieved using the method listed as Algorithm 6.

Adaptive intersection management algorithms such as the ITSP (see Section 3.2 of Chapter 3) can utilise such freedom in choosing the combination of flows to activate. Throughput is maximised by activating those combinations of flow directions that will result in maximising the intersection throughput or optimising other intersection management criteria.

6.5.3 Infrastructure simulation

The simulator can simulate variable message signs (VMS), induction loops and ITS-Sensors. All of those entities can be placed on lanes (see appendix A) and interact with the traffic.

Induction loops are assumed to be robust sensors (see Subsection 2.2.3 of Chapter 2) and therefore able to detect all the vehicles passing above them without failures. The ITS-Sensors, relying on visual observations are prone to measurement errors. It is assumed that the location of a vehicle is measured with accuracy of δ_x and its speed is measured with accuracy δ_V. The measured speed $\hat{V}$ and location $\hat{x}$ are therefore given by:

$$\begin{aligned} \hat{V} = &\quad V + \delta_V \\ \hat{x} = &\quad x + \delta_x \end{aligned} \tag{6.23}$$

Where V and x are the true values of vehicle speed and location. The measurement errors δ_x and δ_V are randomised form the range of $\delta_x \in [-2, 2]$ meters and $\delta_V \in [-0.5, 0.5]$ m/s.

6.6 Defining simulation scenarios

The simulator is required to be configured prior to carrying out simulation studies. The configuration includes loading an appropriate simulation scenario. A simulation scenario consists of a defined traffic network, boundary conditions and simulator engine parameters. The traffic network definition comprises roads and intersections definitions.

Currently only cross-type intersection definitions are implemented, but the modular architecture of the simulator will allow to add more types in the future. A cross type intersection can have four or less intersecting roads, and each road can have a virtually unlimited number of lanes in either direction. Intersection definition includes its location in two dimensional space and information about which turns are available from which direction.

Once the basic traffic network topology has been defined it is possible to place dynamic entities such as vehicle spawners, traffic signs, sensors and individual vehicles. Vehicle spawners inject vehicles at the position they are placed. The starting speed of the injected vehicles is user defined. The injection rate of the vehicle can either be fixed or injection plans can be defined to enable variable, time dependant injection rates. The configuration of the traffic simulator is discussed in detail in Appendix A.

6.7 Gathering simulation data

The traffic simulator software collects traffic data from various sources and different scopes. The collected traffic data has been divided into the following categories:

- **Route data**

  Route data consists of information about all routes traversed by the vehicles in a given scenario. Each vehicle records information about its journey, including the time it spent queued and waiting. Throttle and brake usage history is recorded as well and is used to estimate fuel that has been consumed by each vehicle. When exiting the simulated area the vehicle's journey data is retrieved and sent to the Area Management Service (AMS), which is responsible for organising the route data information received from the vehicles. When the simulation is finished the route information is downloaded from the AMS and saved in the simulation scenario directory.
- **Flow data**

  Flow data is collected by both lane sensors and vehicles on each road section. The vehicles provide information on the time it took them to traverse a given road section as well as the corresponding wait time, queue time and energy consumption. The lane sensors provide information about the amount of vehicles that traverse a given section in a given time period. The collection of flow data enables the identification of road sections which are prone to congestion. Flow data is used by the AMS to perform dynamic routing (see Section 3.6 of Chapter 3).

- **Signal timing data**

  In order to provide insight on how various ICA switch traffic lights, information on signalling phase duration is recorded.

6.8 ITS-Cloud integration

The integration with the ITS-Cloud was the main reason for the development of a new traffic simulation tool rather than use of the existing tools.

All the intersections were equipped with the cloud interface and software necessary to request the allocation of ICS. The lane sensors were implemented in such way that a static ITS-Cloud service is created to allow other services to read sensor data. Similarly the simulated V2I nodes create an RCS.

The simulator can use the ITS-Cloud to link multiple simulator instances together. This is done to address the limitations of the size of the area that can be simulated on a single computer.

The simulator, besides providing services to interact with the traffic or read its status, can use the ITS-Cloud to up-scale the simulated area. The size of the simulated area is constrained by the computational resources of the machine running it. The simulator can use the ITS-Cloud to mitigate those constraints by splitting the simulated area into smaller sections and simulating each one on a different processing nodes. The process of scaling the simulated area is transparent to the CTMS and is regarded by the traffic control system as one traffic network. The ITS-Cloud enables a service referred to as the Link-Model (LM) to handle the connection and synchronisation between two simulator instances (see Figure 6.14).

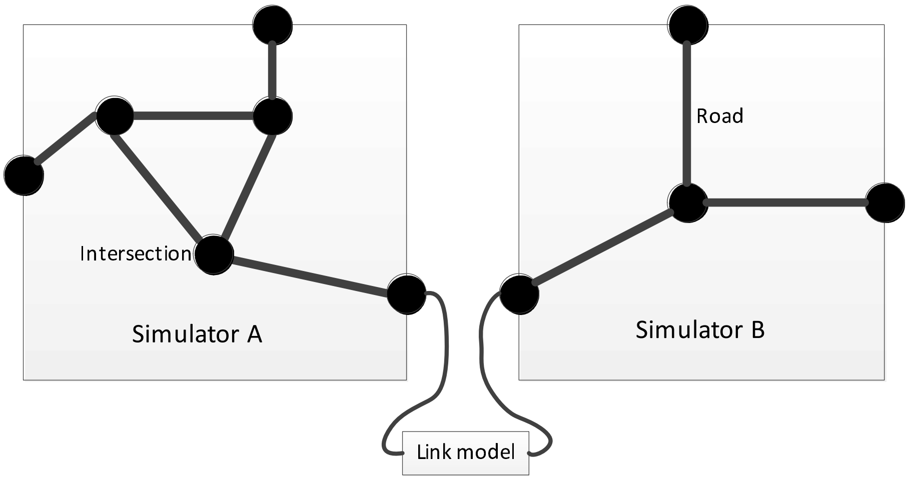


Figure 6.14: The Link-Model can be used to link traffic realms simulated by two different simulator instances and allow vehicles to pass between them.

The LM simulates a road in a macroscopic scale. It was not meant for long corridor simulations but for joining adjacent areas, therefore several simplifying assumptions were made. It is assumed that the vehicles do not change lanes and keep a constant speed while traversing the macroscopic road representation of the link model. The LM was therefore implemented as a set of first in first out (FIFO) queues, one for each lane of the road. A predefined traversal time of T_{LM} is used. When a vehicle enters the LM it is placed at the back of the FIFO queue and its scheduled departure time is noted. The LM service monitors the vehicle at the front of the queue and once its scheduled departure time comes it is removed from the queue and sent to an appropriate simulator instance. The LM passes all the information required to re-create the microscopic scale representation of the vehicle, including its travel time and energy usage data, between the simulator instances.

6.9 Conclusions

The traffic simulation tool described in this chapter provides an environment for deployment, testing and validation of the Cloud based Traffic Management System. While there are several powerful traffic simulation tools available (see Section 2.3 of Chapter 2) none of them was deemed to be suitable for modification allowing for integration with the ITS-Cloud. The new simulation tools, whilst having less features than mature software such as *Vissim*, provides a highly configurable and realistic traffic simulation environment.

The main advantages of the developed simulator include:

- **Nanoscopic aspects**

  Introducing nanoscopic level of detail allows investigation of a greater amount of details regarding the vehicle's behaviour. It allows the modelling of vehicle dynamics and more advanced behaviour such as deceleration by coasting. Simulating nanoscopic aspects of the vehicle models enables the simulations of cooperative adaptive cruise control (CACC) which relies on the vehicle low level controller to maintain the desired speed. These complex behaviours enable each vehicle in the simulation to contribute towards creation of the total energy consumption and $C0_2$ emissions estimate.
- **Scalability**

  The capability to use the ITS-Cloud to scale the simulation helps to mitigate the traffic network size constraints imposed by the computational complexity required by the nanoscopic level of modelling. Provided that sufficient amount of computational nodes is available, very large traffic networks can be simulated.
- **Portability**

  The traffic simulator was implemented from scratch in Java without dependence on any licensed software components. The tool can be run on many platforms, and have been tested on Linux and Microsoft Windows. The instances of the simulator can cooperate within one system regardless of the platform they are running on.

- **Modular design**

  Significant emphasis has been put on the modularity of the simulator design. It allows for the software to be extended with new functionality without having to change its existing components.

The biggest disadvantages of the new simulation tool have been identified as:

- **Computational costs**

  Modelling vehicle components at a nanoscopic scale results in increased processing power and memory demand. Although it is mitigated by the possibility to scale the simulation using the ITS-Cloud to many processing nodes the simulator is constrained to simulating up to 1000 vehicles simultaneously on a standard desktop computer.

- **Specialisation**

  It was realised from the beginning of the project that it would be impossible to integrate all the functionality of a mature traffic simulator (see Section 2.3 of Chapter 2 within the project time frame. The simulator was purpose-designed and implemented to be part of the CTMS. Only the functions essential to the intended application were implemented.

- **Lack of microscopic intersection modelling**

  The vehicles inside the intersections are not modelled in micro scale in the current version of the simulator. The vehicles are not aware of vehicles on incompatible traffic flows and therefore there is no yielding rules implemented. The traffic lights are relied upon to ensure that there will be no collisions within the intersection area.

- **No modelling of unsignalled intersections**

  Due to the lack of microscopic modelling of the area inside an intersection it is impossible to simulate unsignalled intersections.

- **Simplified vehicle model**

  The vehicle model used was a trade-off between computational costs and the depth of simulated details. A simplified representation of the vehicle components such as the engine and brakes was used and the remaining powertrain components were not modelled.

- **Unsophisticated user interface**

  The simulator was designed to collect traffic data from large scale simulations and the emphasis was placed on data collection. A simple graphical user interface was designed to enable visual inspections of the current traffic situation and control the simulation. However many features still remain inaccessible through the interface and require either source code change or editing the scenario configuration files.

Chapter 7

Simulation studies

This chapter summarises the features of the components and the algorithms described in this thesis and demonstrates the effectiveness of the novel elements through simulation studies.

The simulation studies focus initially on evaluating the ITS traffic management components and algorithms that are part of the CTMS. The investigated ITS components include Cooperative Adaptive Cruise Control (CACC), Intersection Approach Trajectory Optimisation (IATO), Dynamic Routing (DR), Micro and Meso Scale Prediction Services (MiSPS and MeSPS). The intersection management algorithms include FC, ILC, ITSP and Two-Step.

The second stage evaluates the ITS components that can cooperate, see Table 7.1 for an outline of such inter-component compatibility. CACC platooning and DR

Table 7.1: ICA features and compatibility with the CTMS system components

ICA	Adaptive	CACC aware	IATO	MiSPS	MeSPS
FC	No	No	Yes	No	No
ILC	Yes	No	No	No	No
ITSP	Yes	Yes	No	No	Yes
Two-Step	Yes	Yes	Yes	Yes	Yes

are independent from the intersection control algorithms (ICA), therefore the choice of ICA does not affect their behaviour. However the ICA can affect performance of the aforementioned components. The ultimate aim of this chapter is therefore to quantify the interactions between all the components of the overall system.

The chapter is organised as follows:

- The first section describes the simulation set-up including the investigated traffic scenarios.
- The second section provides an overview of the criteria used to evaluate the investigated traffic management techniques.
- The next three sections evaluate the non-novel, but required, ITS components namely ACC, CACC, IATO and DR.
- Sections 7.6 and 7.7 investigate the performance of MiSPS and MeSPS, the Micro and Meso scale prediction mechanisms.
- In Section 7.8 the parameter sensitivity study is conducted, where the impact of various tuning parameters used within the CTMS components is investigated.
- Section 7.9 investigates the effect of processing and network communication within the cloud on the traffic management.
- Section 7.10 contains extensive studies of the novel Two-Step traffic management method to evaluate its performance in different scenarios compared to ITSP and benchmark algorithms.
- The final section discusses the findings and concludes the chapter.

7.1 Simulation set-up and scenarios

A set of representative scenarios was designed to evaluate the strengths, weaknesses and robustness of the developed traffic management strategies. A scenario includes an urban area comprising roads with fixed or time dependent speed limits and traffic flows as well as sensors to monitor the traffic and V2I communication capabilities. The traffic flows are calculated based on individual vehicle movement.

Each vehicle is randomised based on a validated family saloon car in terms of both dynamic characteristic and driver behaviour (see Subsections 6.3.2 and 6.3.8 of Chapter 6). The remainder of this section describes the simulation set-up starting with the ITS-Cloud platform components.

7.1.1 ITS-Cloud platform

The ITS-Cloud platform was configured to satisfy the computational needs of all scenarios as well as to demonstrate the portability of the platform. The cloud environment consisted of three heterogeneous computing nodes, two 64 bit Linux machines and one 32 bit Windows machine (see Table 7.2).

The set-up was done in MIRA Ltd. offices. The computers were located in different buildings, therefore corporate network with measured average latency of 4 milliseconds was used for communication.

7.1.2 Urban areas

Four urban areas were selected to capture a range of realistic road layouts within a city boundary including a grid city layout, one of the urban traffic corridors in Coventry, UK an arterial road and a single intersection. A range of speed limits were investigated based on inner city (urban) as well as outer ring roads (suburban) speed limits. In all cases it was assumed that all intersections are managed using

Table 7.2: Test bed configuration

Name	Operating system	CPU model	CPU clock	RAM
UniCalc	Linux 3.7.10	Intel Core i5 M450	2.2 GHz	3 GB
WS1398	Windows 7 (6.1.7601)	Intel Core i5 M560	2.67 GHz	4 GB
LinuxDev1	Linux 3.7.10	Intel Xeon 5120	1.6 GHz	2 GB

traffic lights and vehicles drive on the right hand side.

Speed limits

There were two speed limits considered for the simulated scenarios:

- Urban speed limit

  The urban speed limit was set to $15 m/s$ ($54 km/h$ or $33.75 mph$) which is close to the official urban speed limits in many countries.
- Suburban speed limit

  The suburban speed limit was set to $30 m/s$ ($108 km/h$ or $67.5 mph$) which was meant to represent the speed limit on multi-carriageways in the UK.

Traffic flows

The traffic flow intensities considered for the scenarios are based on the traffic flow intensities observed on a major road in Coventry, UK, namely the northbound Foleshill road (see Figure 7.1).

The heaviest traffic was observed in the morning between 8 am and 9 am and in the afternoon 5 pm and 6:30 pm. The morning peak registered traffic flow intensities between 1000 and 800 vehicles per hour. The flow intensity in the afternoon peak was between 800 and 900 vehicles per hour. The investigated road is a major transit arterial in Coventry therefore the traffic intensity remained high throughout the day varying between 650 and 800 vehicles per hour. The fastest increase in traffic flow intensity was observed between 7:30 am and 8 am where the measured flow increased from about 400 to 1000 vehicles per hour.

- High traffic flow intensity

  The high traffic flow intensity profile is different for each scenario. In the Coventry scenario (see Subsection 7.1.5) the vehicle injection rate was set to 900 vehicles per hour. It corresponds to the morning and afternoon peak hours.

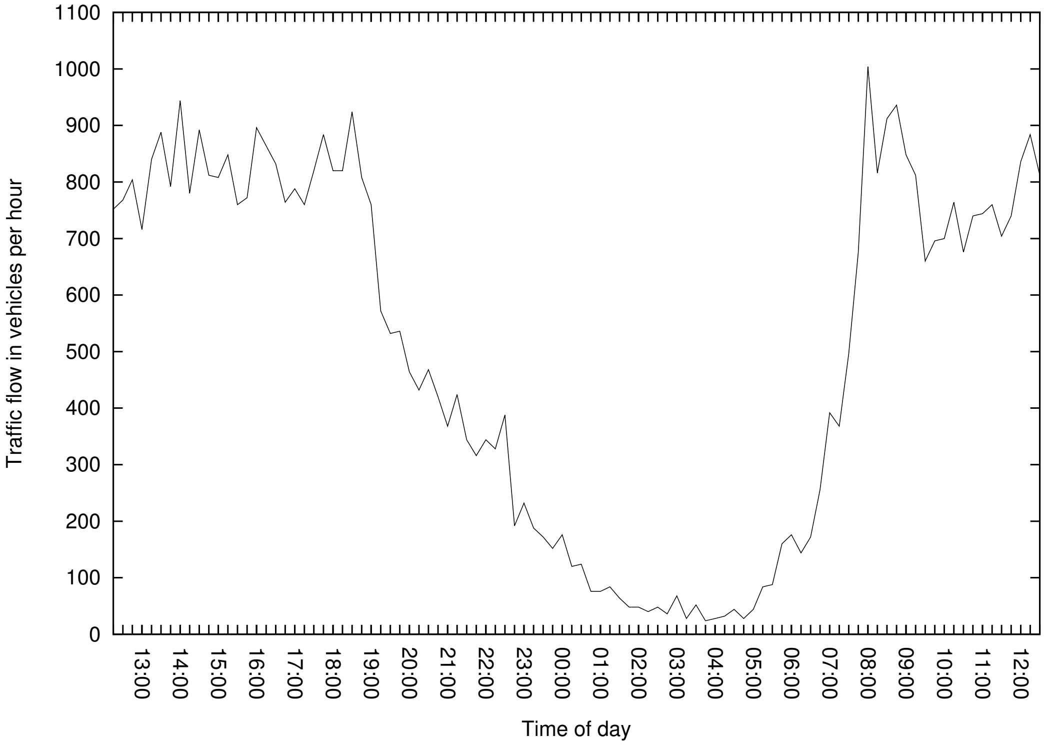


Figure 7.1: Traffic flow measured in Floeshill Road, Coventry, UK between 12:00 29/11/2012 and 12:00 30/11/2012.

In the remaining scenarios the high traffic flow intensity scenario corresponds to a situation where the road network is on the verge of saturation assuming that the intersections are managed using FC and there are no ITS-vehicles present (0% ITS-vehicle concentration rate).

- Low traffic flow intensity

  The low traffic flow intensity assumed injection rate of 200 vehicles per hour per lane.

- Variable traffic flow intensity

  The variable traffic intensity profile varies the vehicle injection rates in time. It assumes the simulation lasts for two hours. During the first 20 minutes the injection rate is constant at 200 vehicles per hour and corresponds to the low traffic intensity profile. During the next 20 minutes, between the 20^{th} and 40^{th} minute of the simulation, the injection rate linearly increases to reach the levels of the high traffic intensity profile. This corresponds to the increase rate observed in Coventry (see Figure 7.1). High traffic intensity conditions are then kept for the next 30 minutes until the 70^{th} minute of the simulation time. In the next 30 minutes, between the 70^{th} and 100^{th} minute of the simulation, the traffic flow intensity decreases linearly back to the levels of 200 vehicles per hour and remain at this low level for an additional 20 minutes.

- Saturated traffic
  In the saturated traffic profile the vehicle injection mechanism was configured to inject a new vehicle every time the previous one has moved away 10 meters from the injection point.

7.1.3 Long corridor

The long corridor scenario consisted of a single lane road and was used in the evaluation of the ACC and CACC platooning mechanisms. The road was of sufficient length to prevent the vehicles from reaching its end throughout the duration of the experiments.

7.1.4 Single intersection scenario

The single intersection scenario aimed to help with evaluation of the CTMS components in saturated traffic conditions, in particular the impact of speed limit and cooperative platooning on intersection throughput.

The scenario comprises two intersecting one way single lane roads. The vehicle injection on both roads was configured to create saturated flow conditions.

7.1.5 Coventry scenario

The Coventry scenario represents a section of the Foleshill road in Coventry, UK. It consists of a main road intersecting with several side roads (see Figure 7.2). All the roads are single carriage ways with urban speed limit.

This scenario aims to evaluate the developed traffic control strategies in a realistic urban environment. Its creation was motivated by obtaining real traffic flow readings

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

Figure 7.2: The Coventry scenario topology.

from an induction loop sensor, which was then used to set the vehicle injection rates in this scenario (see Figure 7.1). The injection rates on the main road can be set according to one of the traffic intensity profiles described on page 137.

The traffic flow intensity measured on the side roads has shown to never grow over 200 vehicles per hour during the measurement time. The worst case of value of 200 vehicles per hour is therefore used on the side roads in this scenario.

The signalling stages in on Foleshill road were configured in north-south / east-west manner. In such configuration turning vehicles must yield to oncoming traffic even though they have green light. The lack of microscopic intersection modelling capability in the simulator means the vehicles are unaware of traffic from the opposing direction making it impossible to replicate the stage configuration from Coventry. Instead of using north-south / east-west signalling stages configuration the intersections are set up to have one stage for each flow direction.

7.1.6 Arterial road scenario

The Arterial Road scenario comprises three lanes in each direction in a suburban environment. It contains four signalled intersections where side roads cross the arterial road. The side roads comprise one lane in each direction (see Figure 7.3). The distance between the intersections varies from 350 to 500m. On the main road 90% of the vehicles were set to travel the entire length of the arterial with the remaining 10% picking a random destination. Half (50%) of the side road traffic was set to cross the arterial and the other half to pick a random destination, joining the arterial.

The previously described Coventry scenario consisted only of single carriageways. Using a scenario with multi lane roads allows for the traffic flow to express behaviours such as lane changing that could not be observed in the previous scenario. The aim of this scenario is to enable the evaluation of the CTMS traffic management methods

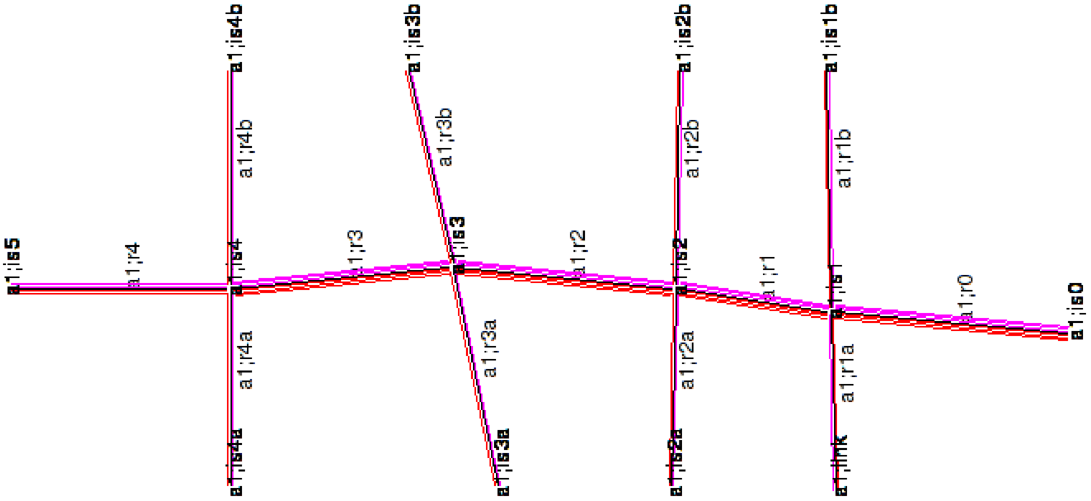

a1;is4b
a1;is3b
a1;is2b
a1;is1b
a1;r4b
a1;r3b
a1;r2b
a1;r1b
a1;is5
a1;r4
a1;is4
a1;r3
a1;is3
a1;r2
a1;is2
a1;r1
a1;is1
a1;r0
a1;is0
a1;r4a
a1;r3a
a1;r2a
a1;r1a
a1;is4a
a1;is3a
a1;is2a
a1;link

Figure 7.3: The arterial road scenario topology.

in more complex environment than allowed by the previous scenario.

The Arterial Road scenario is configured by default for suburban speed and high traffic intensity vehicle injection profile. 930 vehicles are injected per hour on each lane on both ends of the arterial road, resulting in the total injection rate of 2790 vehicles per hour in each direction on the arterial road. The injection rate on the side roads was set to 600 vehicles per hour.

7.1.7 Grid City scenario

The Grid City scenario aimed to resemble an arterial layout of a modern city. The scenario uses a 3 by 4 grid layout and consists of three parallel avenues with three lanes in each direction in north-south orientation and four one way links intersecting with the avenues, two in east to west direction and two in the opposite direction. The distances between intersections are between 500m and 1000m.

This scenario aims evaluate of all traffic management techniques used in the CTMS. The scenario configuration enables all considered vehicle behaviours to occur. Multi lane roads allow vehicles to change lanes and overtake and the topology of the scenario makes it possible to use different paths to reach any destination enabling the evaluation of the DR component.

On the outlying north-south avenues half of the traffic volume was set to continue through the avenue and the other half was set to diagonally cross the simulated area (see Figure 7.4). Such configuration forces the vehicles to employ appropriate lanes to make the required turns at intersections, thus ensuring that routing related lane changing occurs on approach to intersections.

The traffic intensity in this scenario was set to 900 vehicles per hour on each of the east-west roads. The injection rates on the outlying north-south avenues were set to 1800 vehicles per hour, out of which half of the vehicles aimed to travel the

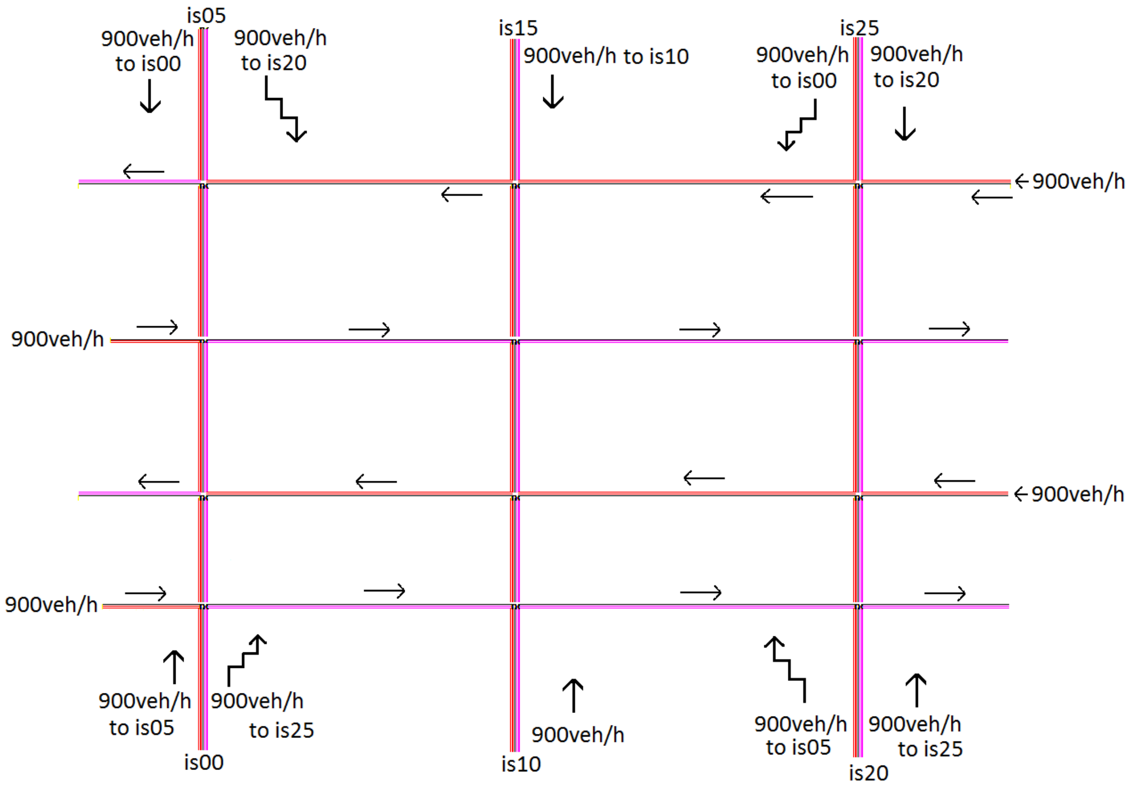

is05
900veh/h
to is00
900veh/h
to is20
is15
900veh/h to is10
is25
900veh/h
to is00
900veh/h
to is20
←900veh/h
900veh/h
←900veh/h
900veh/h
900veh/h
to is05
900veh/h
to is25
is00
900veh/h
is10
900veh/h
to is05
900veh/h
to is25
is20

Figure 7.4: The Grid City scenario topology.

entire length of the road while the other half aimed to diagonally cross the simulated area.

In order to investigate the impact of speed limit on the performance of CTMS and its components the Grid City scenario considered both urban and suburban speed limits.

7.2 Result quantification

The criteria used to evaluate traffic management systems have been identified in Section 2.1 of Chapter 2. In this work, the following widely used criteria were selected:

- **Average journey time**

  The average travel times are expressed as the mean end to end travel time of all vehicles between specified start and finish points. In complex scenarios, such as the Grid City scenario, there are multiple possible routes the vehicles can take to reach the destination. To obtain a common reference which provides the means to compare the performance of the investigated traffic management techniques, a situation where all traffic lights are green is used. All the measured journey times T_J are presented as percentage of such reference that is termed the *base journey time* T_J^{base}.
- **Waiting and queued times**

  Studies have shown that congestion not only negatively impact the economy and the environment but also the driver's mental well-being [6]. The waiting and queued time criteria reflect the time the vehicle had to remain stationary, and had to travel at a reduced speed. The vehicle is considered to be waiting if it travels below 5% of the maximal allowed speed on the lane it is on and it is considered queued if it is forced to travel below 50% of the speed limit by having to follow another vehicle or being unable to change lanes. All waiting and queue time measurements are expressed in terms of percentage of the associated journey time.

- **Energy expenditure**

  The energy expenditure reflects the energy cost of having the vehicle travel to its intended destination. It is derived from the individual vehicle behaviour (see Subsection 6.3.1 of Chapter 6). Similarly to the journey time measures, the energy expenditure E_C measurements are presented relative to the best energy expenditure for each route, which is referred to as *base energy consumption* E_C^{base} within the scope of this work.
- **Throughput**

  The throughput describes the number of vehicles that can pass through a component of the road network in a defined period of time. Throughput is measured in vehicles per hour in this work.

All the time measurements are gathered by the simulator tool (see Section 6.7 of Chapter 6) independently for each defined end to end route, every direction of every road and for the entire simulated traffic network. Throughput is measured for every intersection and road.

7.3 Cooperative platooning evaluation

Cooperative platooning or cooperative adaptive cruise control (CACC) enables vehicles to form tight and stable platoons. It is achieved by communicating the lead vehicle's intent to all the vehicles in the platoon and automatically controlling their speed (see Subsection 6.3.5 of Chapter 6).

The main application area of CACC is motorway traffic and most research in the field was focused there [25, 41, 89, 93, 95, 96]. Some researchers have started to recognise the potential benefit CACC can provide in urban environment with signalled intersections [33, 40].

The aim of this study is to evaluate and compare the behaviour of the implementation of ACC and CACC mechanism used herein. There were four scenarios evaluated. The first scenario investigates the stability of ACC and CACC platoons on a freeway. The second experiment investigates the throughput benefits of CACC in an urban environment. The third evaluates the potential disadvantages of CACC in an urban environment. The fourth experiment evaluates the impact CACC has on journey times in an urban environment. The common assumptions and setting for all experiments are:

- Communication delay of 100 ms for CACC [96].
- Time headway $\tau_{ACC} = 1$ second for ACC platoon and $\tau_{CACC} = 0.6$ second for CACC platoon.

7.3.1 Experiment 1 - freeway

This experiment evaluates vehicles interaction in ACC and CACC platoons on a freeway.

Criteria:

- Separation gap to provide a measure of string stability.

Experiment setup:

- Long corridor scenario, see Subsection 7.1.3.
- A seven vehicle platoon as in [96].

Initial conditions:

- Initial vehicle separations: 10 meters.
- Initial speed set to 20 m/s.

Experiment:

The time headway policy (see Subsection 6.3.5 of Chapter 6) requires the vehicles to

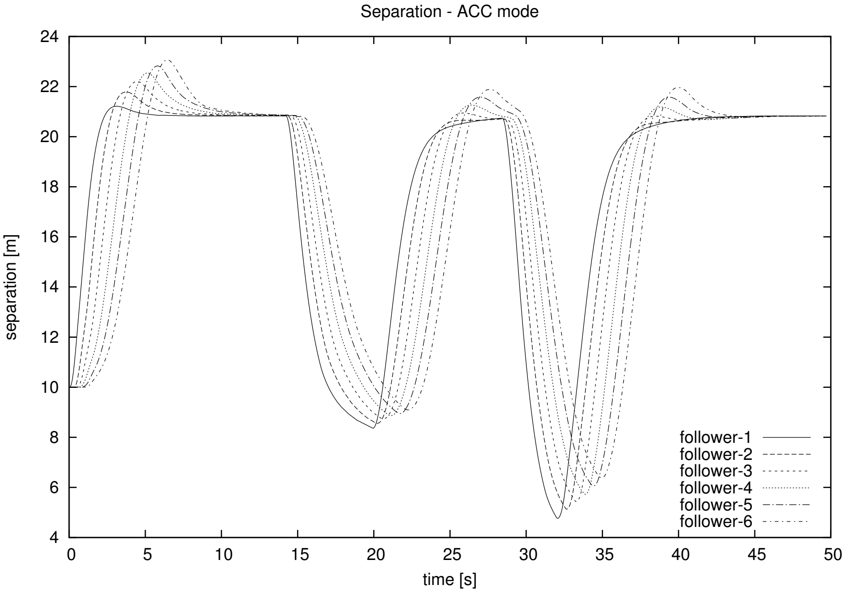


Figure 7.5: Separation gaps between the vehicles in ACC platoon.

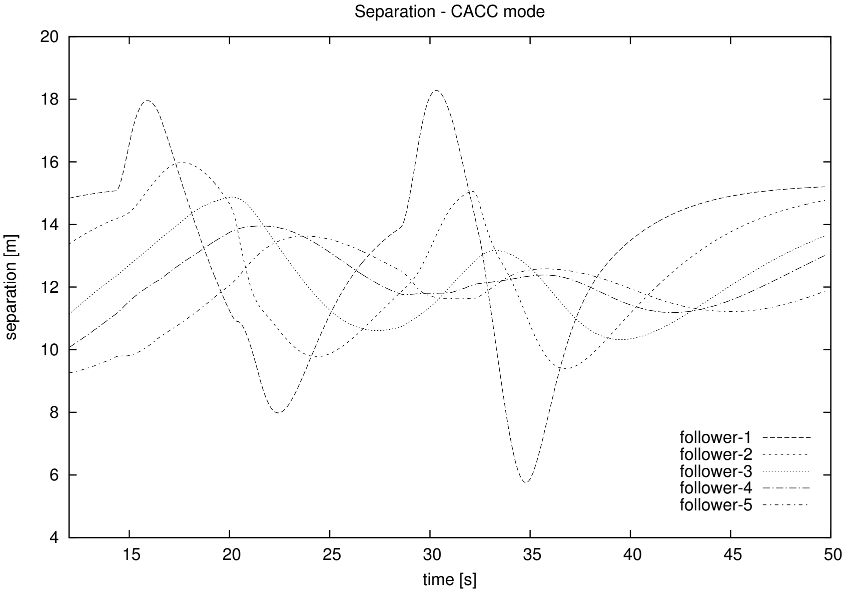


Figure 7.6: Separation gaps between the vehicles in CACC platoon.

maintain a 21 meter separation gap at the initial speed of 20 m/s, forcing vehicles to increase the gap on the beginning of the experiment. The lead vehicle introduces disturbances to the platoon by changing its target speed at specified moments in time (see Figures 7.5 and 7.6). 15 seconds into the experiment the platoon leader starts to decelerate towards the new set speed of 8 m/s, however the manoeuvre is interrupted 5 seconds later as the target speed of the platoon leader is set back to 20 m/s. The next deceleration begins 28 seconds after the simulation started with the lead vehicle aiming to achieve new set speed of 4 m/s. The set speed is set back to 20 m/s in the 32^{nd} second of the simulation.

Results:

- The ACC platoon is shown to be string unstable as the separation gap error grows in each successive vehicle in the platoon during the platoon accelerations (see the increased overshoot in Figure 7.5).
- CACC reduces the separation gap errors, caused by the disturbances introduced by the lead vehicle, as they propagate down the platoon. It is therefore string stable confirming the results in [96]

7.3.2 Experiment 2 - Intersection clear time

This experiment evaluates the benefit of CACC in an urban environment.

Criteria:

- Time it takes for the platoon to clear an intersection.
- Time measurement is started when the lead vehicle crosses the stop line and the stop line crossing time for each vehicle is recorded.

Experiment setup and assumptions:

- Single intersection scenario, as described in Subsection 7.1.4.
- A 23 vehicle platoon.
- The downstream road is kept clear form any obstructions to prevent any

spill-backs that could affect the measurement.

Initial conditions:

- The platoon is stationary in front of the intersection.
- Initial vehicle separations: 1 meter (equal to the required vehicle separation when stationary, see Subsection 6.3.5 of Chapter 6).

Experiment:

- Both ACC and CACC platoons investigated.
- Urban (15 m/s) and suburban (30 m/s) speed limits investigated.

Results:

- The results presented in Figure 7.7 match the theoretical assumptions and previous investigations of platooning mechanisms (see Subsection 2.4.1 of Chapter 2).
- The CACC platoon cleared the intersection 16% faster than the ACC platoon with urban speed limits and 44% faster for suburban speed limit.

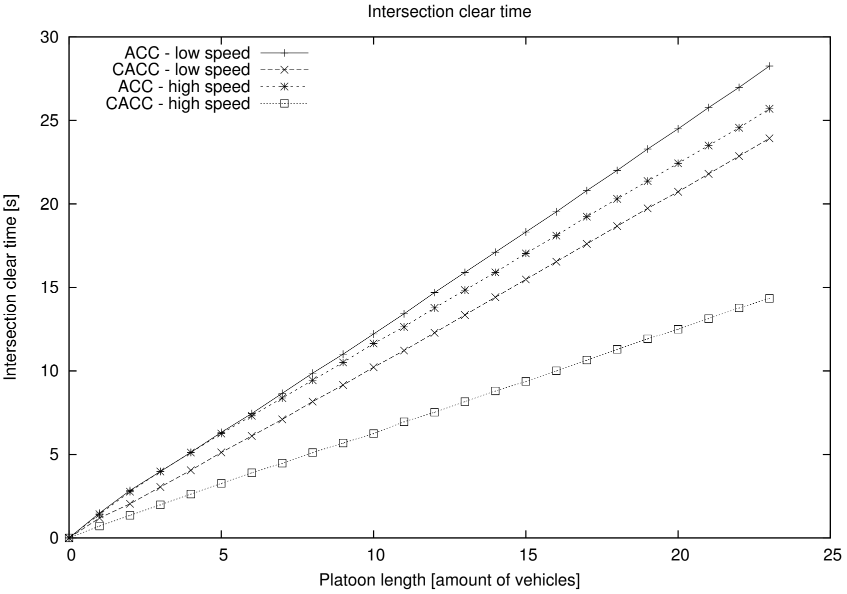


Figure 7.7: Intersection clear times of ACC and CACC platoons in suburban and urban speed limits

7.3.3 Experiment 3 - Impact of traffic lights

The previous experiment demonstrated that CACC platooning can be used to increase road throughput and decrease the time it takes for a platoon to clear an intersection. It was assumed that the traffic lights remain green long enough to let the entire platoon pass. This experiment investigates the impact, in terms of energy consumption, of breaking a platoon of moving vehicles leaving a junction.

Criteria:

- Kinetic energy E_k [kJ] accumulated in each vehicle.
- Cumulated kinetic energy E_k^{sum} [kJ] in all vehicles before the stop line.

Experiment set-up and assumptions:

- Single intersection scenario, as described in Subsection 7.1.4.

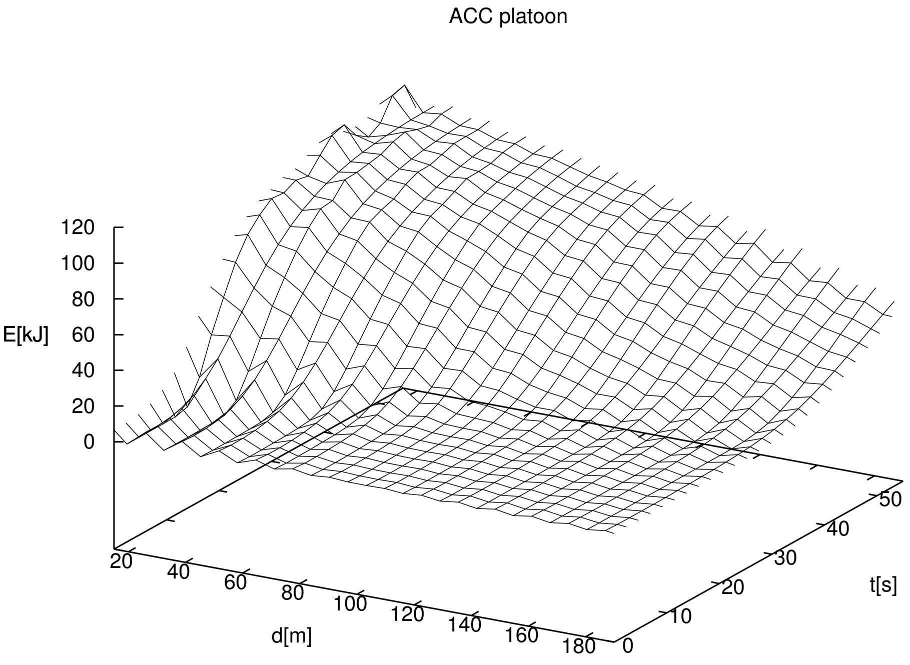


Figure 7.8: Change of kinetic energy distribution in time of an ACC platoon on intersection approach. Initially the platoon is stationary and the light turns green at $t = 0$.

- A 60 vehicle platoon representing a platoon length of 360 meters (when stationary).
- The downstream road is kept clear form any obstructions to prevent any spill-backs that could affect the measurement.

Initial conditions:

- The platoon is stationary in front of the intersection.
- Initial vehicle separations: 1 meter (corresponding to the required vehicle separation when stationary, see Subsection 6.3.5 of Chapter 6).

Experiment:

- Both ACC and CACC platoons investigated.
- Urban (15 m/s) speed limit investigated.

Results:

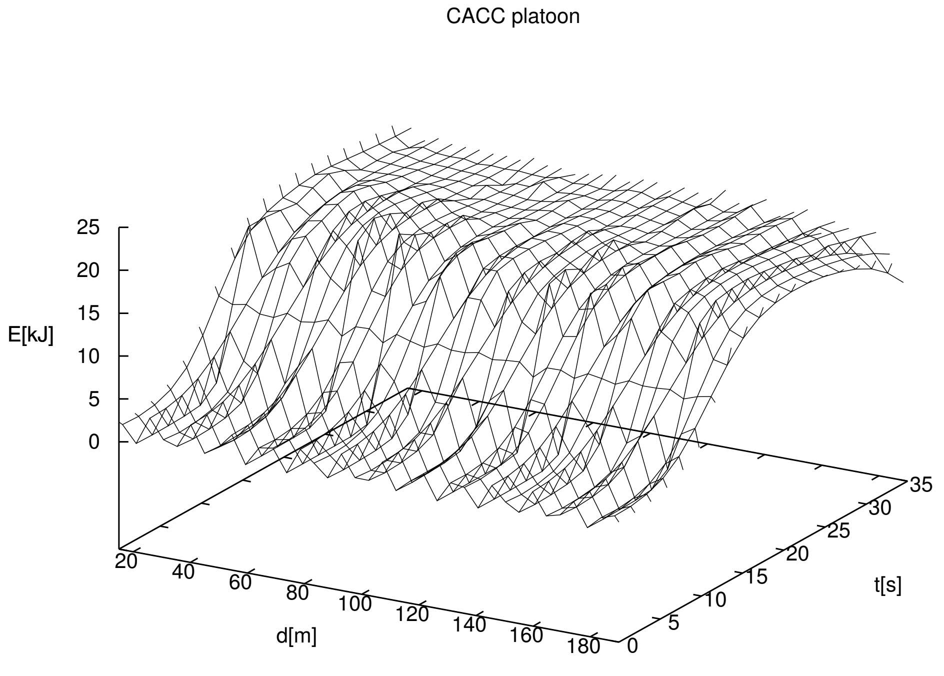


Figure 7.9: Change of kinetic energy distribution in time of a CACC platoon on intersection approach.

Figures 7.8 and 7.9 illustrate the changes in distribution of kinetic energy in ACC and CACC platoons in front of an intersection with respect to the distance from the intersection and simulation time. Figure 7.10 shows the total kinetic energy accumulated on the investigated intersection approach in relation to time that has passed since the traffic light turned green.

It was observed that:

- The vehicles in CACC platoon start moving almost simultaneously. The energy builds up much faster than with ACC, especially in the areas further away from the stop line.
- The vehicles in the ACC platoon start accelerating one after another resulting in a much slower energy build up.
- The kinetic energy cumulated before the stop line E_k^{sum} reduces as the vehicles

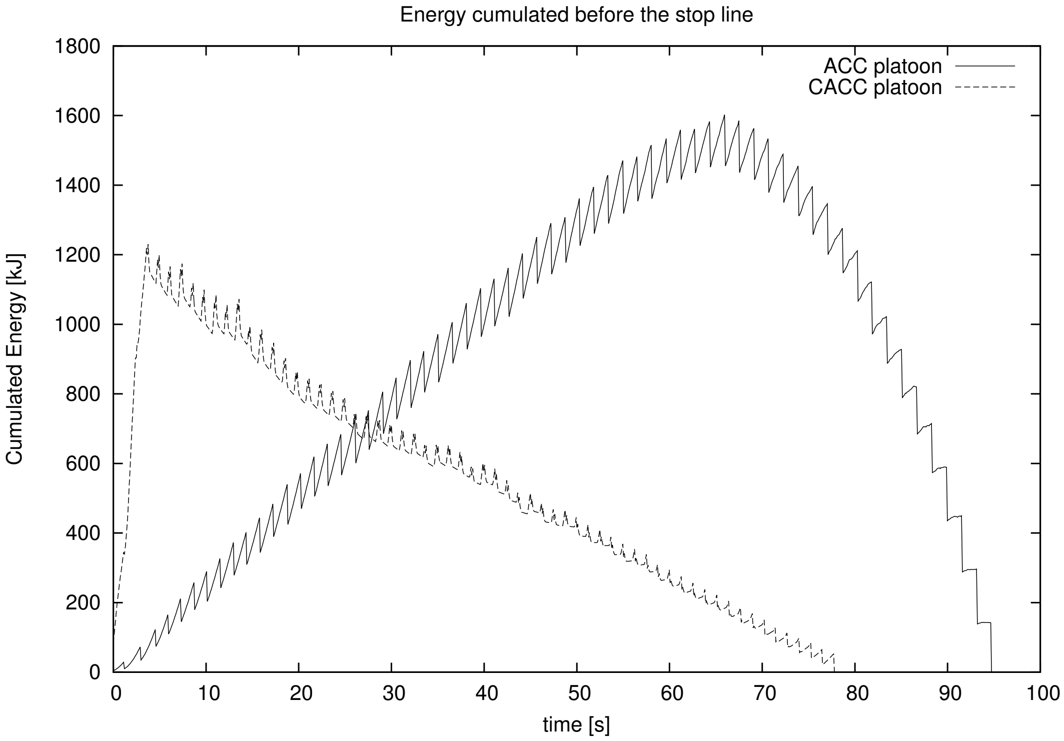


Figure 7.10: The total amount of accumulated kinetic energy on intersection approach.

pass the intersection.

- In the first 25 seconds of the simulation the E_k^{sum} is higher in the CACC platoon.

Conclusion:

If the traffic light turns red before the entire platoon passes the intersection the kinetic energy accumulated in the vehicles that have not managed to pass is lost. If such a change occurs within the first 25 seconds after the platoon started moving more energy would be lost in the CACC platoon than in its ACC governed counterpart. Therefore traffic lights can cause the CACC platooning mechanism to increase energy consumption and CO_2 emissions in an urban environment compared to ACC.

7.3.4 Experiment 4 - Impact on journey times in urban environment

The previous simulations investigated the behaviour of pre-assembled vehicle platoons and assumed that either pure ACC or CACC were used. This experiment investigates the impact of CACC on journey times in an urban environment using ACC, CACC and mixed platooning.

Criteria:

- Average journey time in the entire traffic network.
- Cumulated kinetic energy E_k^{sum} [kJ] in all vehicles before the stop line.

Experiment setup and assumptions:

- Arterial Road scenario, as described in Subsection 7.1.6.
- Variable traffic flow intensity (see Section 7.1 and Subsection 7.1.6).
- Suburban speed limit (30 m/s).
- Only ITS-vehicles can engage in CACC platooning.
- Stage lengths for FC: 32 seconds for the arterial road and to 19 seconds for

the side roads.

- FC and ITSP intersection management methods were used.

Experiment:

- ITS vehicle penetration rates of 0%, 50% and 100% investigated.

Results:

Figure 7.11 shows the length of the average journey time with respect to the simulation time. It was observed that in low traffic intensity (from 0 to 2000 and 5500 to 7000 seconds of the simulation) the CACC platooning does not have any effect on the journey time, regardless of the type of ICA used. This is due to insufficient traffic density to allow formation of CACC platoons [94]. In the heavy traffic conditions however, the CACC was shown to reduce journey times.

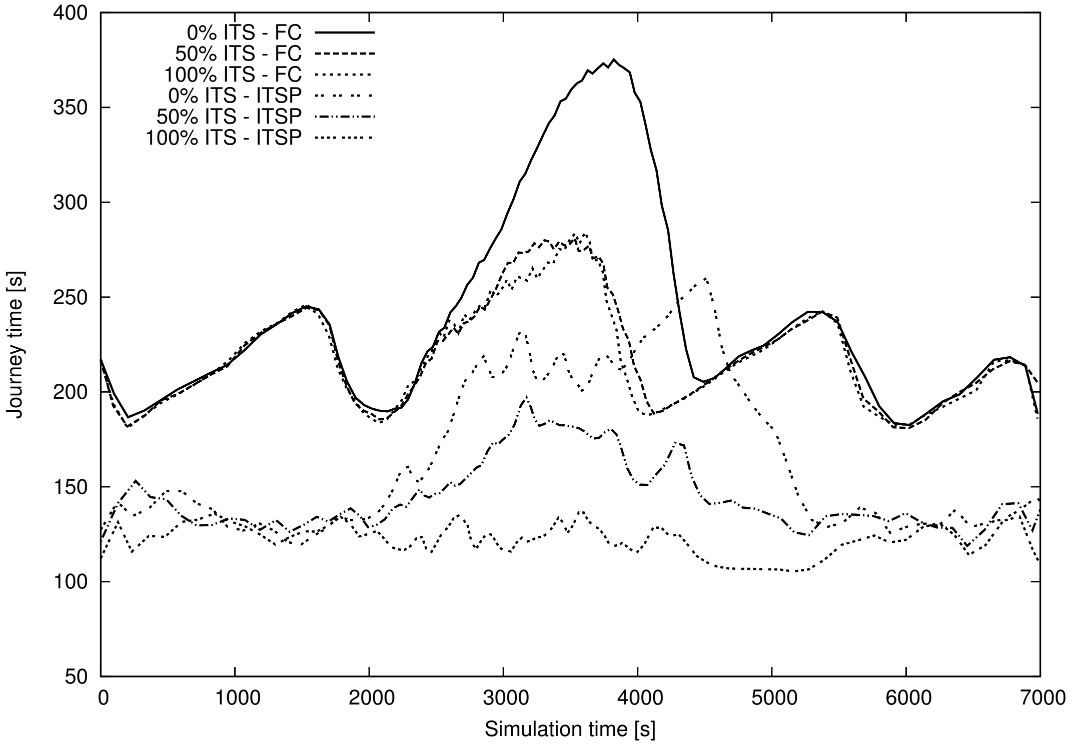


Figure 7.11: Influence of CACC on journey time using variable traffic flow (see Section 7.1)

Conclusion:

In agreement with literature, CACC platooning was found to increase the road network throughput. The performance of CACC is limited by the fixed cycle based intersection management, which is observable as the small difference between 50% ITS and 100% ITS-vehicle penetration rate. Even though the intersection clearance time under CACC (100% ITS-vehicle penetration rate) is much faster, as observed in the previous example, the platoon is still likely to be stopped at the next intersection. Such effect can be mitigated by using an adaptive intersection control scheme such as the ITSP, which will attempt to switch the traffic lights taking into account incoming platoons. Figure 7.11 shows that CACC can increase road throughput to such extent that the journey time is not affected by traffic conditions that would cause severe delays otherwise.

7.4 Intersection Approach Trajectory Optimisation

This experiments evaluates the Intersection Approach Trajectory Optimisation (IATO) component of the CTMS.

Criteria: (see Section 7.2)

- Average journey time T_J as a percentage of T_J^{base}.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- Two scenario maps:
  - Arterial Road scenario, see Subsection 7.1.6.
  - Grid City scenario, see Subsection 7.1.7.
- FC intersection management method used with following stage lengths:
  - Arterial Scenario: 32 seconds for the arterial road and to 19 seconds for

the side roads.

  - Grid city: 20 seconds for each stage.

- High traffic flow intensity (see Section 7.1 and Subsection 7.1.6).
- Suburban speed limit (30 m/s).
- IATO component is evaluated alone, CACC and DR are disabled in this experiment.

Experiment:

- ITS vehicle penetration rates of 0% to 100% in 10% increments were investigated.

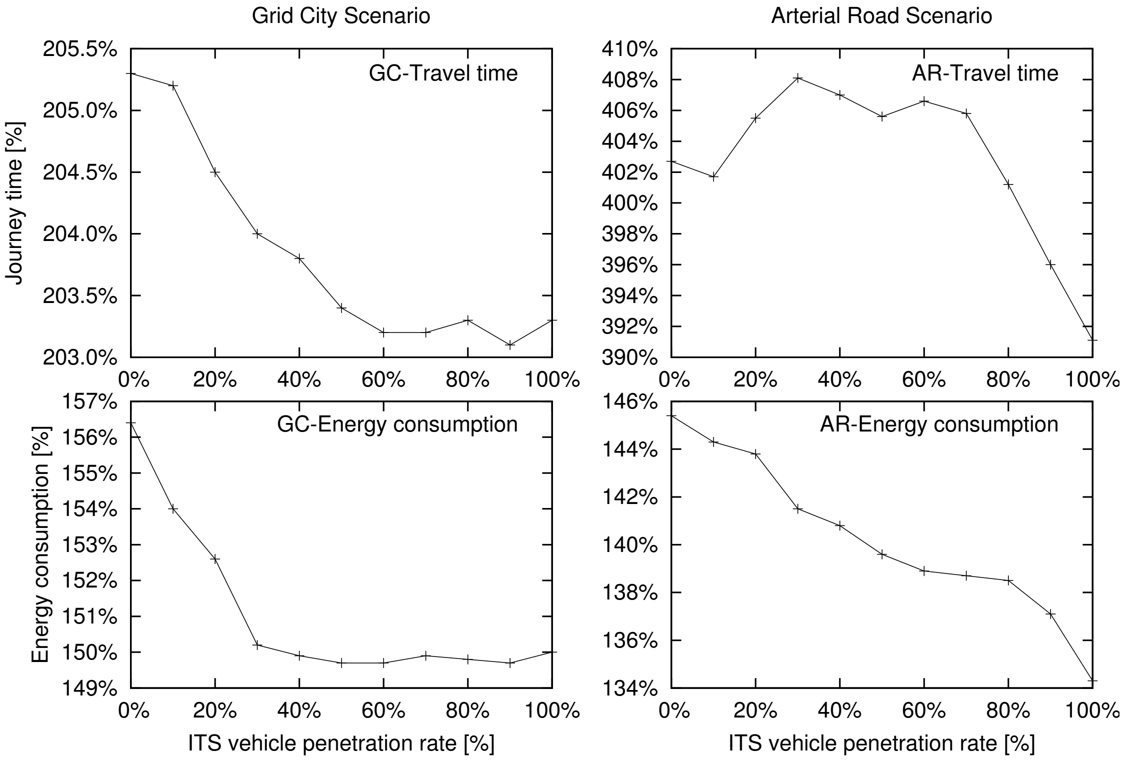


Figure 7.12: IATO performance in different ITS-vehicle concentration rates using FC intersection management method in Grid City and Arterial Road scenarios.

Results:

- It is shown in Figure 7.12 that the E_C reduces with an increasing ITS-vehicle penetration rate. The rate of improvement varies significantly between the scenarios:
    - In the Grid City scenario the biggest E_C reduction of 6.5% E_C^{base} occurs below 30% ITS-vehicle penetration rate whereas only minor improvement can be observed by increasing the number of ITS-vehicles beyond 30%.
    - In the Arterial Road scenario E_C reduces gradually throughout the entire range of ITS-vehicle penetration rate. E_C reduction of 12% E_C^{base} is observed.
- In the Grid City scenario T_J decreases with growing ITS-vehicle penetration rate in range of 0% to 60%. Small T_J reduction of 2% T_J^{base} is observed.
- T_J increase of up to 5% T_J^{base} was observed in the Arterial Road Scenario in ITS-vehicle penetration rates between 20% and 70%. A decrease of 11% T_J^{base} is observed with 100% ITS-vehicle penetration rate.

Conclusions:

- IATO is capable of reducing both energy consumption and journey times confirming observations made in [10].
- The observed differences in energy consumption improvement rates in the investigated scenarios are caused by normal vehicles overtaking ITS-vehicles that optimise their approach, therefore moving slower than the speed limit allows. In Grid City the vehicles are forced to follow a particular lane in order to reach their destination, therefore no spontaneous lane changing occurs just before the intersection. This forces normal vehicles to follow ITS-vehicles when they optimise the intersection approach trajectory. In the Arterial scenario the vehicles can assume any of the three lanes in the arterial road enabling normal vehicles to overtake the ITS-vehicle in the vicinity of the intersection. This

behaviour persists until all normal vehicles are eliminated with a 100% ITS vehicle penetration rate.

- The increase in T_J observed in the Arterial scenario is due to strategy used within IATO to estimate the distance the vehicle is going to travel before the traffic light switching event occurs (see Section 3.4 of Chapter 3). If the vehicle enters the IATO managed area with a speed significantly lower than the speed limit it might be advised to decelerate even though it could still make through the intersection if it accelerated. It is impossible however to determine if reaching the velocity required to make it through is possible unless the vehicles in front of the vehicle in question participate in a CACC platoon. Owing to short distances between intersections in the Arterial Road scenario, the vehicles enter the IATO range while they are still gaining speed after slowing down at the previous intersection. The result is reduced traffic stage utilisation leading to increased T_J in situations where ACC vehicles mix with CACC vehicles as shown in Figure 7.12.

7.5 Dynamic routing

This experiment evaluates the Dynamic routing (DR) component of the CTMS.

Criteria: (see Section 7.2)

- Average journey time T_J as a percentage of T_J^{base}.
- Average waiting time T_W as a percentage of T_J.
- Average queued time T_Q as a percentage of T_J.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- Grid City scenario, as described in Subsection 7.1.7.
- FC intersection management method was used. Each stage duration was set

to 20 seconds.

- High traffic flow intensity (see Section 7.1 and Subsection 7.1.6).
- Suburban speed limit (30 m/s).
- DR component is evaluated alone, CACC and IATO are disabled in this experiment.
- DR calculates the routes using information on road congestion and traffic flow as described in Section 3.6 of Chapter 3.
- Only ITS-vehicles are able to receive DR advice.

Experiment:

- ITS vehicle penetration rates of 0% to 100% in 10% increments were investigated.

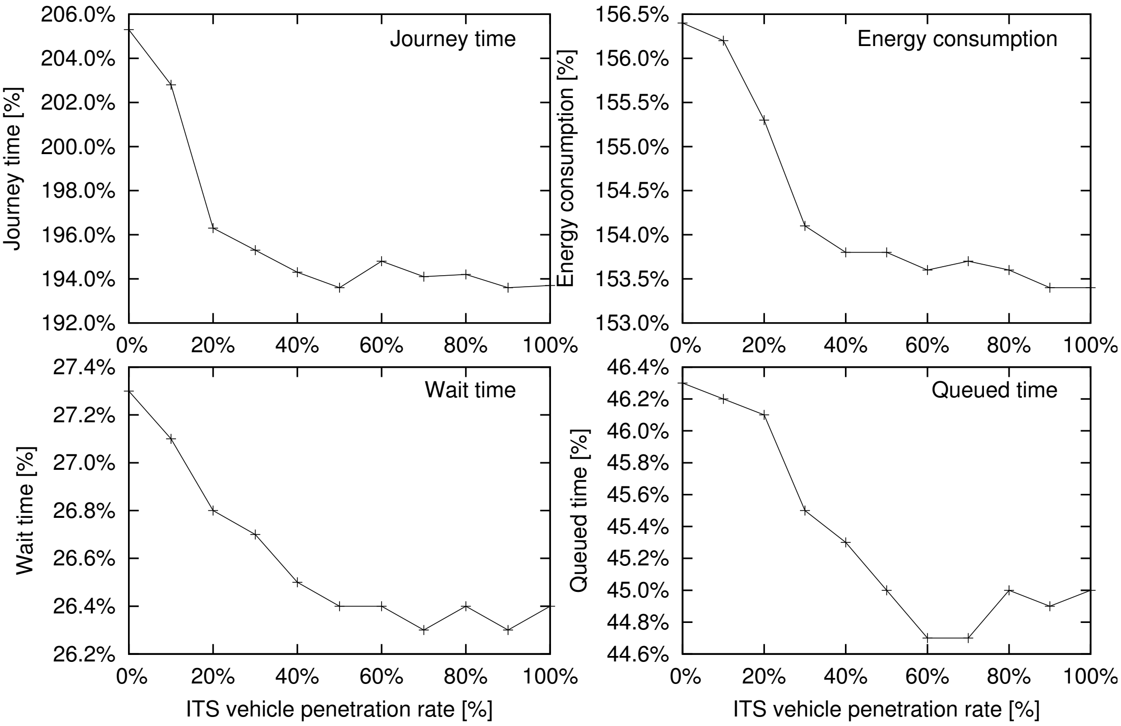


Figure 7.13: DR performance in different ITS-vehicle concentration rates using FC intersection management method in Grid City scenario.

Results:

- Dynamic routing has shown to improve all the considered criteria (see Figure 7.13).
- The biggest improvement rate is observed in the range of 0% to 20% ITS-vehicle penetration ranges.
- The vast majority of improvement is achieved for 50% ITS-vehicle penetration rate with further improvement being minor.
- The average journey time T_J decreased by 9% T_J^{base} and the average energy consumption E_C was reduced by 3% E_C^{base}.

Conclusion:

- The DR mechanism has shown to be an effective mean for improving traffic flow in an urban environment.
- Similarly to IATO, the DR is effective even in low ITS-vehicle penetration rates.

7.6 The Micro Scale Prediction Service

This experiment evaluated the performance of the Micro Scale Prediction Service (MiSPS).

Criteria:

- Accuracy of prediction measured in terms of error between actual and predicted traffic flow.

Experiment set-up and assumptions:

- Arterial Road scenario, as described in Subsection 7.1.6.
- Variable traffic flow intensity (see Section 7.1).
- Suburban speed limit (30 m/s).
- The Two-Step intersection management method was used to enable MiSPS.

- ITS sensors provide the following information:
  - Vehicle location with the accuracy of $\delta_x = 2$ meters.
  - Vehicle speed with accuracy $\delta_V = 0.5$ m/s.
  - Vehicle ITS capability.

Experiment:

- In order to examine similar amount of ITS and normal vehicles, the ITS-vehicle penetration rate was set to 50%.
- The performance of the component was evaluated by comparing the predicted situation images to the situation image snapshots taken at the appropriate moments in time. Assuming that the prediction was made H_p time units into the future and was based on a situation image $S_i(t)$ observed at time t the obtained predicted situation image $\hat{S}_i(t + H_p|t)$ will be compared to the situation image $S_i(t + H_p)$ observed at time $t + H_p$.
- The predicted and observed situation images are compared using the ITS pressure indexes π_i defined in Section 3.2 of Chapter 3.

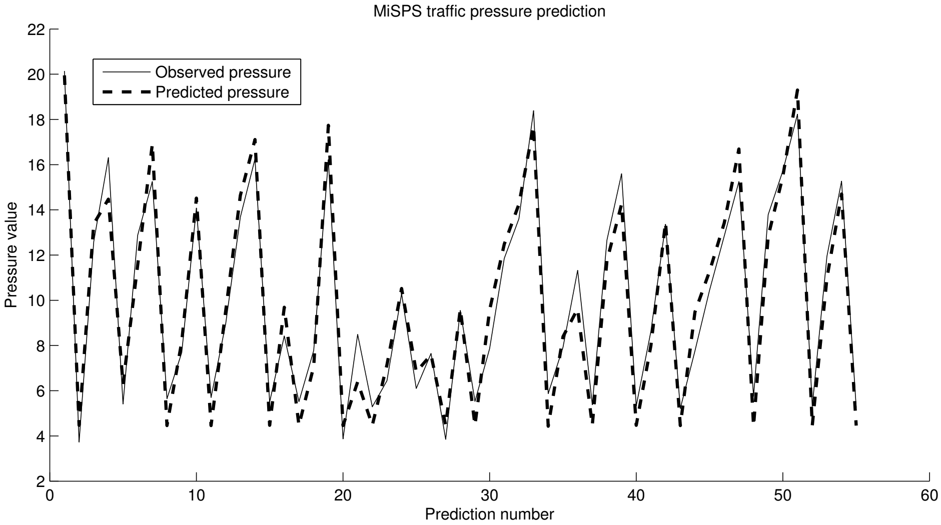


Figure 7.14: Accuracy of MiSPS prediction observed on road r1 approach to intersection is2 in the Arterial Road scenario (see Figure 7.3).

Results:

- In the whole scenario the predictions have shown to be of a very good quality yielding the root mean square error (RMSE) of 0.99. In the ITS pressure index the value of 1 corresponds to a single stopped vehicle (it increases with the vehicle's speed as described in Section 3.2 of Chapter 3)
- The coefficient of determination RT^2 of 76% was obtained.
- The observed pressure fluctuations observed in Figure 7.14 are due to changes in traffic conditions caused by the traffic lights. When the traffic light is red the pressure builds up resulting in the ICA eventually switching the lights to green. When that happens the pressure starts to drop until it is deemed that the traffic light should switch to a different stage.

Conclusion:

The quality of the MiSPS method was deemed satisfactory, however the obtained result is subject to several implementation assumptions. MiSPS uses the same micro simulation mechanism that is used to simulate the entire traffic network, therefore the vehicle model used in the prediction might be too similar to the one used in the simulation representing the real world. However, to make the prediction more realistic the following approximations were implemented:

- Vehicle model tuning differences

  The model parameters of vehicles in the simulation are randomised upon creation of the vehicles making every vehicle slightly different (see Section 6.3 of Chapter 6). The MiSPS uses a single set of tuning parameters for all vehicles in the predicted situation image.
- Added sensor measurement errors in terms of vehicle location and speed

  The accuracy of the situation image depends on the quality of data obtained from the ITS-sensors, which are prone to measurement errors (see Subsection 6.5.3 of Chapter 6).

- Routing and lane selection

  The vehicles may have to choose a certain lane in order to follow their desired route. The information about each vehicle's destination is not part of the situation image therefore the MiSPS is not able to predict the lateral behaviour of the vehicles.
- Driver behaviour

  The driver remains the most unpredictable component of a vehicle control loop [144]. MiSPS uses the same driver model for all the vehicle in the prediction whilst, in the road network simulation, the parameters describing the driver are randomised (see Subsection 6.3.8 of Chapter 6).

Note that if such a method is used in practise it will be possible to increase the model accuracy by accounting for the typical range of vehicles likely to be involved using information from ITS sensors or upstream junctions.

7.7 The Meso Scale Prediction Service

This experiment evaluates the Meso Scale Prediction Service (MeSPS) component of the CTMS which predicts the influence of surrounding traffic on a particular intersection.

Criteria:

- Accuracy of prediction measured in terms of error between the predicted and measured amount of vehicles.

Experiment set-up and assumptions:

- Grid City scenario (see Subsection 7.1.7) modified as follows:
  - Nine intersections in 3 by 3 grid layout and single carriageway roads.
  - The intersections were labelled using matrix indexing where *is00* was in the south-west corner and *is22* was in the north-eastern corner.

  - Variable vehicle injection rate was used using the same rates as in the original Grid City scenario (see Subsection 7.1.7).
  - ITS vehicle penetration rate of 50% was investigated.
  - The link between middle intersection (*is11*) and its northern neighbour (*is12*) was chosen for investigation.
- Variable traffic flow intensity (see Section 7.1).
- Suburban speed limit (30 m/s).

Experiment:

- To test the self tuning capability of MeSPS the proportions of vehicles taking different routes changed in time. Initially 80% of vehicles travelling north from *is10* through *is11* would continue north towards *is12* with the remaining 20% randomly taking either west or east turn. Throughout the simulations the proportions changed linearly to 40% vehicles continuing north and 60% taking a different turn.

Results:

Figure 7.15 shows the predicted amount of vehicles on the investigated link plotted against its measured equivalent.

- The initial probability of a vehicle choosing the northbound link from *is11* was 33% (see Section 5.4 of Chapter 5), while the real rate was 80%. This is visible on the plot as the estimator initially underestimates the amount of arriving vehicles (observations 1 to 6).
- The quality of the prediction improves as the estimator tunes itself.
- The heavy traffic conditions occurring between the 28^{th} and the 37^{th} estimation result in large differences between the estimated and measured vehicle counts. This is caused by the traffic queuing at *is12* and obstructing the lane sensor.

Conclusions:

- The overall performance of the MeSPS mechanism is satisfactory, with the prediction error usually being constrained to a few vehicles.

- The validation mechanism that is used to tune the predictor is prone to measurement issues caused by congested traffic. Owing to this observation the self tuning component is prevented from changing the tuning parameter of the predictor if more than 60 vehicles in a wave were predicted and the error between the measured and predicted situations is greater than 20%. This is done to prevent losing the current predictor parameters by attempting to retune it using invalid input data.

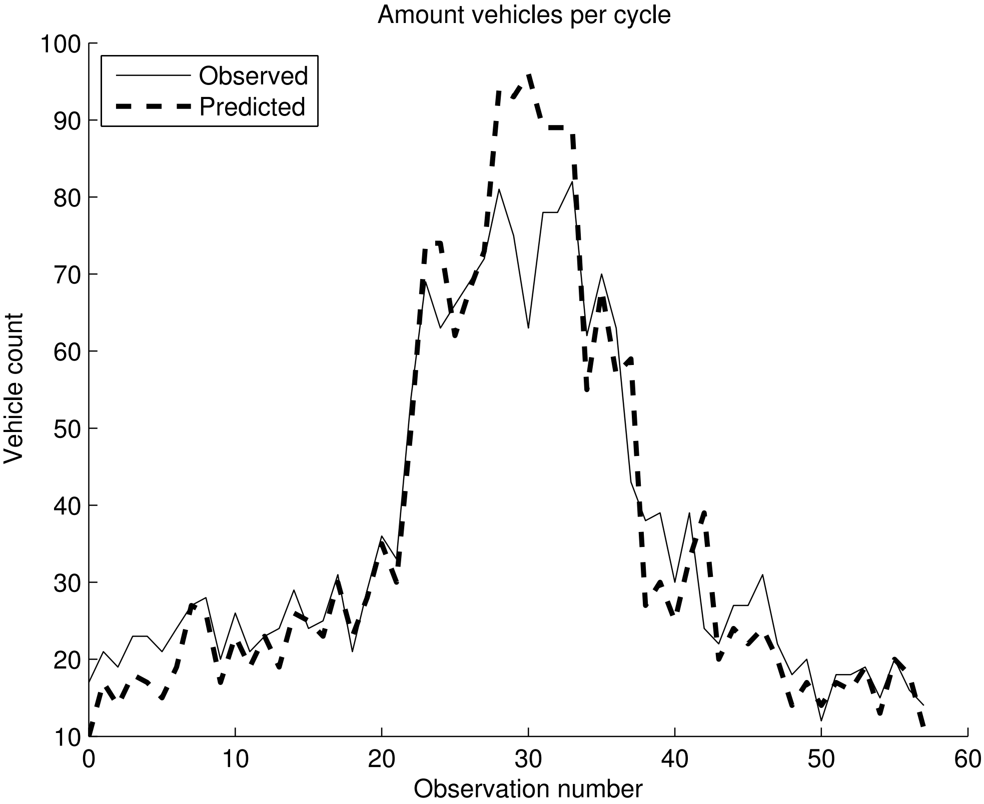


Figure 7.15: The amount of vehicles predicted by MeSPS to arrive at is12 from is11 plotted against validation measurement.

7.8 Parameter sensitivity

This section investigates the sensitivity of the traffic management components tuning parameters. The following criteria were used in the first three experiments:

- Average journey time T_J as a percentage of T_J^{base}.
- Average energy consumption E_C as a percentage of E_C^{base}.

The experiment set-up and assumptions common to the first three experiments were:

- Use of Coventry scenario (see Subsection 7.1.5).
- Variable traffic flow intensity (see Section 7.1).
- Urban speed limit (15 m/s).

7.8.1 Experiment 1 - ICA sampling rate

This experiment investigates the impact of intersection control algorithm (ICA) sampling rate on the traffic management performance.

Experiment set-up and assumptions:

- ITS vehicle penetration rate of 50% was investigated.
- Two different intersection management methods were used: ITSP and Two-Step.

Experiment:

- Sampling rate value range of $T_{ICA} = [0.5, 2.5]$ seconds were used with 0.5 second increments.

Table 7.3: Impact of ICA sampling rate

		ICA sampling rate T_{ICA} [s]				
		0.5	1.0	1.5	2.0	2.5
ITSP	T_J	131.9%	126.8%	130.8%	131.6%	144.3%
	E_C	143.9%	143.7%	140.8%	142.0%	142.3%
Two-Step	T_J	133.3%	134.0%	139.9%	144.3%	146.1%
	E_C	137.1%	135.6%	134.3%	138.0%	140.3%

Results:

The simulation results are shown in Table 7.3.

- Sampling too slowly reduces the traffic management efficiency and increase E_C and T_J.
- The Two-Step method was designed to handle network transmission latency (see Section 3.3 of Chapter 3). The additional latency incurred by low sampling rates can be handled in a similar manner making the Two-Step less sensitive to T_{ICA} changes than ITSP. ITSP requires timely implementation of the traffic management decision in order to be effective.

Conclusion:

The value of $T_{ICA} = 1s$ was selected as a good compromise solution between the bandwidth usage and the performance of both investigated intersection management methods. It has shown to be a good choice for both algorithms and such value was used for all the remaining experiments in this work.

7.8.2 Experiment 2 - ITSP speed weight

This experiment investigates the impact of the speed coefficient weighting parameter β_V used in the ITSP and inherently in the Two-Step ICA. It is used by the ITSP to determine the impact the vehicle speed has on the constructed pressure coefficient (see Section 3.2 of Chapter 3).

Experiment:

- ITS vehicle penetration rate of 50% was investigated.
- ITSP and Two-Step intersection management methods used.
- $\beta_V = [0, 0.3]$ with an increment of 0.05 was investigated.

Results:

The simulation results are shown in Table 7.4.

- Under ITSP intersection control T_J increases and E_C decreases with increasing

Table 7.4: Impact of ITSP speed weighting β_V

		ITSP speed weight β_V						
		0	0.05	0.1	0.15	0.2	0.25	0.3
ITSP	T_J	125.4%	126.8%	127.9%	132.2%	132.8%	139.4%	141.0%
	E_C	146.4%	143.7%	141.8%	141.1%	140.2%	139.7%	139.9%
Two-Step	T_J	134.7%	135.4%	131.3%	132.8%	132.6%	133.6%	135.5%
	E_C	136.2%	135.6%	133.6%	132.5%	132.9%	134.2%	134.2%

β_V. The increase in T_J remains low for $\beta_V = [0,\ 0.1]$ and accelerates thereafter. On the other hand the decrease of E_C in this range is more significant than in the remaining β_V range.

- The Two-Step has shown to be less sensitive to the parameter change than ITSP. This is due to the effect of IATO diminishing the incorrect choice of β_V. The best E_C and T_J were achieved with $\beta_V = [0.1,\ 0.2]$.

Conclusion:

The most appropriate value of the speed coefficient weighting parameter is $\beta_V = 0.1$.

7.8.3 Experiment 3 - the time headway

This experiment investigates the impact of the time headway τ_{ACC} used in ACC platooning has on E_C and T_J.

Experiment:

- ACC controlled vehicles (ITS vehicle penetration rate of 0%) were investigated.
- ITSP and Two-Step intersection management methods used.
- Based on time headway investigations in [90, 94, 145] the parameter range of $\tau_{ACC} = [0.6,\ 1.2]$ seconds was chosen for investigation.

Results:

Table 7.5 shows that both T_J and E_C increase with increasing τ_{ACC}. Increase in T_J is caused by the reduced road throughput due to increased inter vehicle gaps. An

Table 7.5: Impact of time headway τ_{ACC}

		Time headway τ_{ACC}						
		0.6	0.7	0.8	0.9	1.0	1.1	1.2
ITSP	T_J	119.3%	121.3%	122.7%	125.4%	126.8%	133.8%	143.8%
	E_C	136.5%	138.1%	139.1%	142.6%	143.7%	147.7%	152.9%
Two-Step	T_J	121.4%	121.8%	124.1%	128.4%	135.4%	143.9%	159.8%
	E_C	127.4%	127.5%	127.5%	131.0%	135.6%	138.3%	144.7%

increase in E_C is observed due to increased amount of time vehicles have to stop at traffic lights. The vehicles have to stop on traffic lights more often increasing energy consumption.

Conclusion:

Adopting lower time headways increases traffic performance in terms of T_J and E_C. However setting the time headway too low might result in string instability [145] and potentially hazardous situations. The experiment has led to adopting the lowest value of τ_{ACC}=1 second considered in [94]. Such value was subsequently used with ACC platooning mechanism in all experiments conducted in this work.

7.8.4 Experiment 4 - Speed limit

This experiment evaluates the impact of the speed limit on road network throughput with vehicles under ACC and CACC control.

Criteria:

- Intersection throughput measured in vehicles per hour.

Experiment set-up and assumptions:

- Single intersection scenario, as described in Subsection 7.1.4.

Experiment:

- ACC (0% ITS-vehicle penetration) and CACC (100% ITS-vehicle penetration) platoons were investigated.
- Intersection management methods used: FC, ITSP and Two-Step.

- Speed limits from 10 m/s to 30 m/s in 5 m/s increments were investigated.

Table 7.6: Speed limit impact on throughput [veh/h]

		Speed limit [m/s]				
		10	15	20	25	30
FC	ACC	2200	2602	2804	2933	3007
	CACC	2723	3355	3955	4485	4847
ITSP	ACC	2214	2615	2821	2980	3075
	CACC	2616	3041	3534	4079	4633
Two-Step	ACC	2266	2646	2861	3002	3085
	CACC	2635	3092	3688	4266	4788

Results:

- It was observed that with ACC platoons increasing speed limit from $10m/s$ to $20m/s$ resulted in a throughput increase of 26% to 27% depending on the type of ICA used. Similar results were obtained in [96] where an throughput increase of 28% was observed.
- The advantage CACC holds over ACC in terms of throughput grows with increasing speed limit. In the $10m/s$ speed limit such advantage is within the range of $16 - 24\%$ and in the $30m/s$ speed limit the advantage grows to $55 - 61\%$, depending on the type of ICA used.
- Using CACC with the FC intersection control yielded better results than using the adaptive intersection control techniques. This is caused by the reduced number of traffic light switches performed by FC than by the adaptive ICA. The FC stage length was set to be 50 seconds and the average duration of a stage with the adaptive ICA varied from 30 to 40 seconds.

7.9 Cloud system performance

In this section the consequences communication delays incurred by using a networked, distributed processing environment are assessed. Use of the ITS-Cloud, despite bringing many advantages, may introduce significant communications overheads that can affect the traffic management performance. The communication delays in CTMS can be divided into two categories: processing and network delays. Network delays occur in the cloud system due to its distributed nature which requires the nodes to communicate through a network.

Processing delays are due to the time it takes to process the data in the cloud instead doing it locally, which also includes network transmission delays. In CTMS each client specifies an update interval when connecting to a Sensor Service (SS), which defines the SS sampling rate. The update interval of 500 milliseconds was chosen to provide decent refresh rate while keeping the bandwidth usage low. This means that on average 250ms delay is applied to each sensor reading even without counting the network delays. Due to the same reasons the ICS run the traffic optimisation cycle (which includes execution of ICA) in a one second interval. The time it takes to run each optimisation cycle is negligible compared to the sampling rate.

7.9.1 Impact of data processing latency

This experiment evaluates the impact of data processing latency in the CTMS on the performance of traffic management.

Criteria:

- Average journey time T_J as a percentage of T_J^{base}.
- Average waiting time T_W as a percentage of T_J.
- Average queued time T_Q as a percentage of T_J.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- Arterial Road scenario, as described in Subsection 7.1.6.
- Variable traffic flow intensity (see Section 7.1).

- Suburban speed limit (30 m/s).

Experiment:

- ITS vehicle penetration rates of 0% to 100% in 10% increments were investigated.
- Local and cloud versions of ITSP intersection management method were used.
    - The local delay-free configuration of the TMS represents an unrealistic absolute best (delay-wise) situation, where there are no communication or data handling delays in the system. In this test case the traffic management algorithm was run inside the Traffic Simulator application. Sensor readings are accessed in real time and are time-wise accurate to a single sample length. The communication delay between the ICA and the traffic light controller is one sample (100ms) however there are no

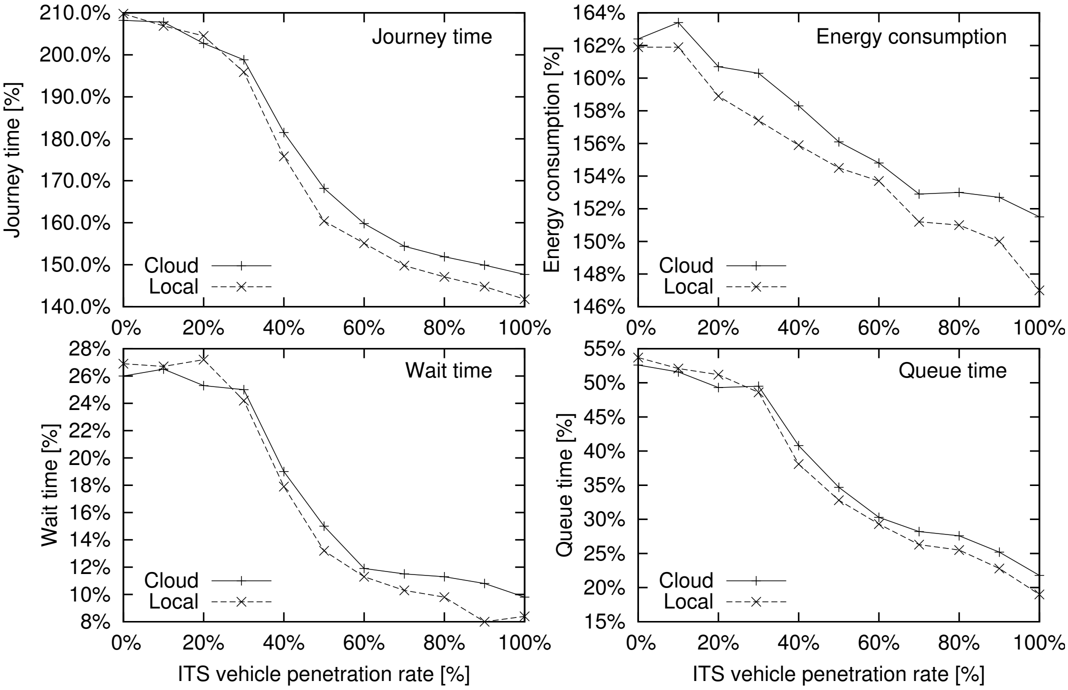


Figure 7.16: Performance comparison between a local implementation of ITSP and its equivalent running in the CTMS.

data handling delays and no bandwidth issues.

  - The remote, cloud based, version of ITSP suffers from delays due to transmission and remote processing latency as described in Section 7.9.

Results:

- As expected, the delay-free version of ITSP outperformed its counterpart that used CTMS in all considered measures. It is worth keeping in mind that the delay-free situation is not achievable in reality.
- The delays associated with CTMS data processing resulted in an T_J increase of up to 7% T_J^{base} and E_C increase up to 4.5% E_C^{base}.
- T_W and T_Q were affected by up to 1.4% and 2.8% respectfully.

Conclusion:

The processing delays affecting the traffic management are noticeable but do not reduce the performance significantly. In reality it would be impossible to implement a delay-free intersection control that was used as the benchmark in this experiment.

7.9.2 Impact of network latency

This experiment investigates the impact of network latency on the CTMS traffic control performance.

A wide area network (WAN) emulator *netem* [146] was used to simulate various quality networks that could form the backbone of the CTMS. The primary interest being the investigation of the impact the network latency has on the performance of the traffic management. Besides the network delay parameters, such as expected value and distribution of transport delay, it can simulate packet loss and many other wide area network parameters.

In order to create an appropriate test scenario, internet latency measurements were performed between the MIRA site and several locations worldwide using an on-line latency measurement tool [147]. Figure 7.17 shows that the latency varies from 5 milliseconds to a different area in the UK to over 280 milliseconds to a server located in China. No packet losses were observed when doing those measurements.

The experiment was conducted as follows:

Criteria:

- Average journey time T_J as a percentage of T_J^{base}.
- Average waiting time T_W as a percentage of T_J.
- Average queued time T_Q as a percentage of T_J.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- Arterial Road scenario, as described in Subsection 7.1.6.
- High traffic flow intensity (see Section 7.1).
- Suburban speed limit (30 m/s).

Experiment:

- ITS vehicle penetration rates of 0% to 100% in 10% increments were investigated.

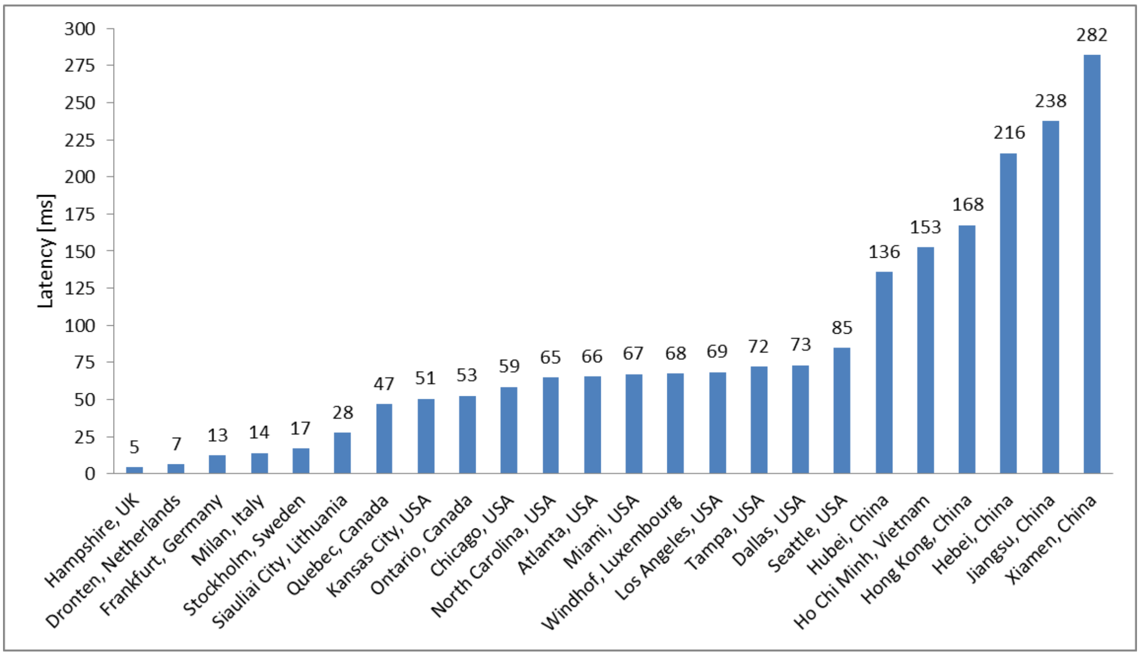


Figure 7.17: Measured latency between mira.co.uk and selected destinations.

- Three network qualities were investigated using *netem*:
  - High quality network with low (5 ms) latency and no packet loss representing a country-local network.
  - Medium quality network with high latency (250 ms) and 1% packet loss rate which meant to reflect conditions on a long distance link such as the connection between the UK and China.
  - Low quality network with very high latency (500 ms) and 2% packet loss rate representing a damaged or congested data link.

Results:

- The CTMS is able to cope even with a low quality network link, however the performance is not left unaffected.

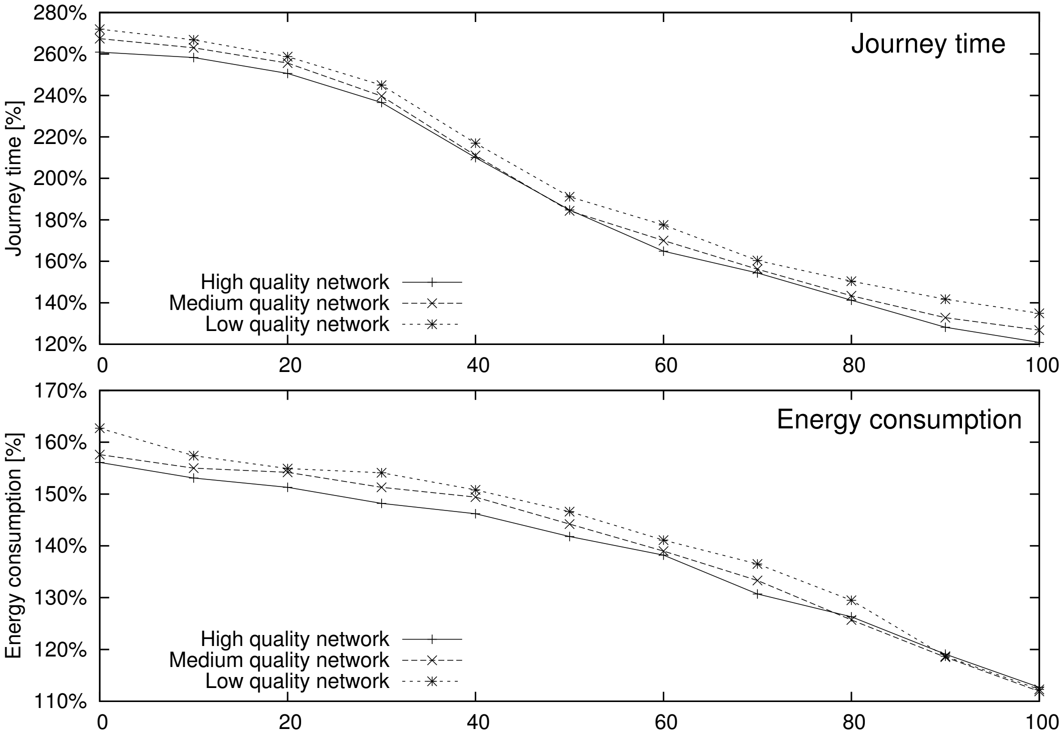


Figure 7.18: Impact of network quality on traffic management performance of the Two-Step method.

- Use of the medium quality network, compared to the high quality network, resulted in performance degradation of T_J up to 6% T_J^{base} and E_C up to 3% E_C^{base}.
- Similarly using the low quality network caused an increase of T_J up to 14% T_J^{base} and E_C up to 6% E_C^{base}.

Conclusion:

In a distributed processing system such as the CTMS the quality of the network connection is crucial to the system performance. The experiment has shown that even though the system is capable of functioning based on low quality network links the traffic management performance is significantly reduced.

7.10 Evaluation of complex traffic management solutions

Previous sections investigated the performance of individual CTMS components. The impact of parameters has been evaluated and their values established. This section evaluates all the CTMS components working together. The main aim is to compare the performance of the novel Two-Step traffic management technique to the remaining traffic management techniques.

Adaptive intersection control approaches have shown to improve traffic flow significantly. In [22] improved traffic flow for a single intersection led to a reduction of the average delays by 11.5% and improved throughput by 0.5%. A larger road network was modelled in [65] using cellular automata. One of the scenarios evaluated a 4 by 4 grid similar to the Grid City scenario used in this work. The average journey times were reduced by up to 8% in heavy traffic and 15% in the light traffic scenario. The adaptive traffic signal control system evaluated in [29] increased the average vehicle speed by 1.5% in the peak traffic and 6.59% in average case. The modelled scenario was an urban grid of 9 by 7 blocks.

The fixed cycles (FC) method is commonly used by authors [22, 23, 29, 35, 65] to evaluate adaptive intersection management schemes. It was noted in [65] that adaptive traffic control almost always yields better results than FC approach and with the adaptive vehicle actuated traffic management schemes becoming mature some authors chose them to benchmark their methods [148, 149, 150]. In this work both approaches were adopted. The novel traffic management scheme was benchmarked using both FC and simple adaptive, vehicle actuated scheme (ILC).

The analysis was divided into three case studies, one for each scenario. The common features of the experiments are:

Criteria:

- Average journey time T_J as a percentage of T_J^{base}.
- Average waiting time T_W as a percentage of T_J.
- Average queued time T_Q as a percentage of T_J.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- ITS-vehicle penetration rate varying from 0% to 100% in 10% increment.
- Each experiment simulated two hours of traffic.

7.10.1 Coventry scenario analysis

This experiment evaluates the novel Two-Step traffic optimisation method in a realistic urban environment modelled after a major route in Coventry, UK (see Subsection 7.1.5).

Experiment set-up and assumptions:

- Coventry scenario, as described in Subsection 7.1.5.
- High and low traffic flow intensities (see Section 7.1).
- Urban speed limit (15 m/s).

Experiment:

- ILC, ITSP and Two-Step intersection management methods used.

Results:

- The simulation results for the high traffic flow intensity profile are shown in Figure 7.19. There following observations were made:
  - ITSP and Two-Step outperformed the ILC method against almost every measured criteria.
  - Even without smart vehicles present ITSP registered journey times of approximately 132% of T_J^{Base} (see Section 7.2), which is just 32% longer than the best possible journey time T_J^{Base} on the investigated route.
  - The journey time decreased in almost linear manner with the increase of ITS vehicle concentration rate, yielding journey times close to 127% of

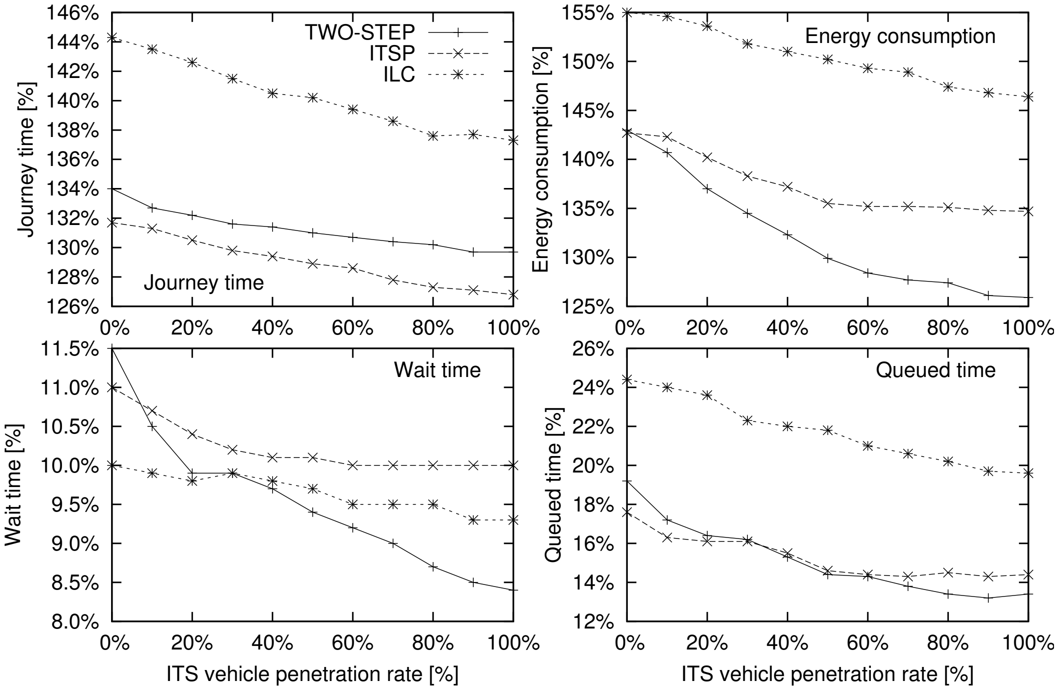


Figure 7.19: Coventry scenario, high traffic density

T_J^{Base}.

- The novel Two-Step technique leads to increases in journey time between 1-3% T_J compared to ITSP. Journey times range from 134% without any ITS-vehicles present to 129% when all of the traffic consists of ITS-vehicles.
- The ILC was outperformed by the pressure algorithms yielding journey times varying from 144% to 137% T_J^{Base}, which was 10% to 12% worse than the journey times under ITSP management.
- The best performance in terms of E_C was achieved using the Two-Step method. The greatest energy efficiency improvement rate was observed in range of 0% to 60% ITS-vehicle concentration rates, with ITSP outperformed by 7% and ILC by 21%. Such improvement can be largely attributed to the IATO, as observed in Section 7.4 of this chapter.

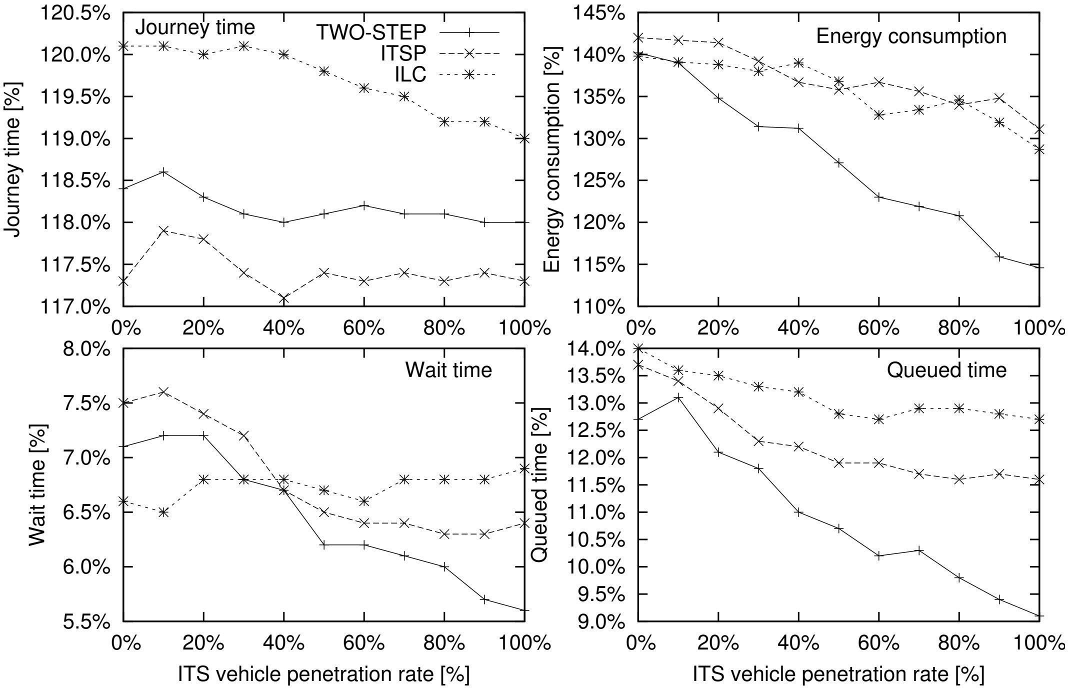


Figure 7.20: Coventry scenario, low traffic density

  - It is worth remembering that both wait and queue times are presented as a fraction of the total journey time under the given circumstances (see Section 7.2). Figure 7.19 shows that due to the CACC platooning waiting T_W and queue T_Q times decreased with increasing ITS-vehicle penetration rates. The rate of such decrease varied depending on the ICA. The Two-Step method reduced the waiting times the most by through the use of the IATO.
- The results obtained using the low traffic flow intensity profile can be seen in Figure 7.20. The observations are as follows:
  - The differences between journey times were much smaller than in the heavy traffic variant of the scenario.
  - The average journey times were approximately 120% T_J^{Base} for the ILC, 117% T_J^{Base} for the ITSP and 118% T_J^{Base} for the Two-Step.
  - It was observed that the ITS-vehicle penetration rate had little impact on T_J. It was observed in [93] that sufficient vehicle density has to be reached in order for the platooning to occur, therefore the benefits associated with CACC over the traffic flow are insignificant.
  - It was observed that the energy consumption E_C was similar in all evaluated ICA in low (0-10%) ITS-vehicle penetration rates. The ILC and ITSP algorithms are shown to produce similar results across the entire examined ITS-vehicle penetration range.
  - Due to the availability of the IATO the Two-Step method demonstrated E_C reduction over the other methods as the concentration of ITS vehicles increased.

Conclusions:

- In both traffic intensity variants of the examined scenario it was observed that the type of ICA is the defining factor for most of the journey time characteristics. The supporting technologies cooperating with ITS vehicles being able to deliver an improvement which is relatively small in comparison. However whilst the type of ICA still has a significant impact, supporting technologies such as the IATO have a considerable contribution towards energy saving.

- In this particular scenario the novel Two-Step optimisation technique was slightly inferior, in terms of journey time, to the ITSP method it was derived from. This was caused by the negative effect of IATO on intersection throughput when the distances between the intersections are too short thereby preventing the vehicles from reaching their cruising speed, see Section 7.4.
- Despite the slight disadvantage of Two-Step against ITSP in terms of journey time, the energy savings delivered by the Two-Step method are significantly larger than those provided by ITSP, especially in high ITS vehicle concentration rates. The improvements associated with the Two-Step are about 10% E_C^{Base} in heavy traffic and 14% E_C^{Base} in light traffic.

7.10.2 Arterial Road scenario analysis

This experiment evaluates the novel Two-Step traffic optimisation method using the Arterial Road scenario (see Subsection 7.1.6).

Criteria:

- Average journey time T_J as a percentage of T_J^{base}.
- Average waiting time T_W as a percentage of T_J.
- Average queued time T_Q as a percentage of T_J.
- Average energy consumption E_C as a percentage of E_C^{base}.

Experiment set-up and assumptions:

- Arterial Road scenario, as described in Subsection 7.1.6.
- High traffic flow intensity (see Section 7.1).

- Suburban speed limit (30 m/s).
- The stage timings for the FC algorithm were set to 32 seconds for the arterial road and to 19 seconds for the side roads.

Experiment:

- Intersection management methods used: FC, ILC, ITSP and Two-Step.

Observations and conclusions:

The simulation results can be seen in Figure 7.21.

- The fixed cycle intersection control outperformed against every criteria the ILC, which is an adaptive ICA. The main reason is that in congested conditions the amounts of stage switch-overs should be minimised by using very long signalling stages [123]. It was measured that ILC activates the arterial road

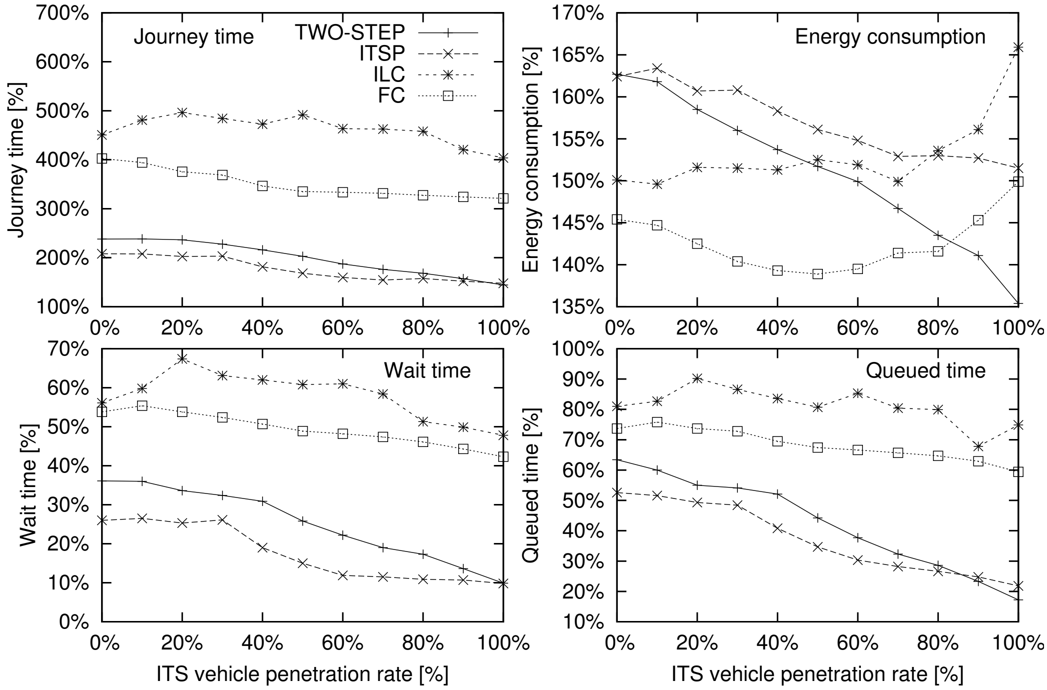


Figure 7.21: The performance of the investigated ICA in high intensity traffic flow conditions in the Arterial Road scenario.

stages for 14 seconds on average, while FC was configured to keep those stages active for 32 seconds.

- The frequent stage switches performed by ILC resulted in the network becoming saturated. This revealed a vulnerability in induction loop based vehicle detection which does not provide the vehicle speed as well as the loop occupation. Assuming that the induction loop sensor can only count vehicles, a slow moving line of vehicles in congested conditions can be confused with sparse traffic. The ILC algorithm estimates the time it would take to clear the vehicle queue, and when activating the stage it assumes that all vehicles will make it through the intersection (see Subsection 3.1.2 of Chapter 3). Such assumption makes the ILC vulnerable to spill-backs from downstream intersections that could prevent the vehicles from crossing the intersection. Using just inflow induction loops the system is unable to verify that the entire queue has been cleared.
- ITS-sensors can observe the traffic in real time and unlike induction loops are not affected by traffic spill-backs. It is visible that the pressure based algorithms handled such congested traffic conditions well, reducing T_J from 402% T_J^{Base} (FC) to 238% T_J^{Base} (Two-Step) and 208% T_J^{Base} (ITSP).
- A significant increase in energy consumption in high (above 70%) ITS-vehicle concentration rates was observed when the traffic was governed by the CACC platoon unaware ICA such as FC and ILC (see Table 7.1 and Figure 7.21). The phenomenon is caused by inappropriate switching of traffic lights leading to wasted energy in CACC platoons, see Subsection 7.3.2. Such a negative effect can be partially mitigated by using a CACC platoon aware ICA (ITSP and Two-Step). It is impossible however to completely avoid platoon breakup by the intersection controller. There are situations, where a pressure based ICA will still decide to switch cycles while a platoon is in transit. Such a situation can occur when there is another platoon, with greater pressure (more cars,

greater speed), approaching form an other direction. The amount of wasted energy can however be reduced by using the Two-Step method with an IATO mechanism to advise the vehicles of the imminent signal change.

7.10.3 Grid City scenario analysis

The Grid City is the largest and most complex scenario. Contrary to the previous scenarios, there is no single major flow direction. There are multiple sets of journey start-end points configured, each defining a distinct route, which can overlap and interact with other routes in different sections of the scenario. There are several different routes a vehicle can take to reach its destination, and with the help of the Dynamic Routing subsystem the CTMS endeavours to improve traffic flow by suggesting alternative routes to ITS-vehicles.

The experiment was carried out in following manner:

Experiment set-up and assumptions:

- Grid City scenario, as described in Subsection 7.1.7.
- High traffic flow intensity (see Section 7.1).
- Urban (15 m/s) and suburban (30 m/s) speed limits.

Experiment:

- FC, ILC, ITSP and Two-Step intersection management methods used.

Results:

- Figure 7.22 shows the simulation results for the Grid City scenario under heavy traffic conditions and high speed limit. The following observations were made:
  - The Two-Step management technique performed well, yielding significant improvement in energy consumption while retaining journey time performance of the ITSP. Similarly to previously investigated scenarios the energy expenditure decreased with an increase of ITS vehicle penetration, however the rate of such improvement and the range of ITS vehicle penetration

rate was different. In all investigated ICA studies a significant improvement in journey time and energy consumption is observed in low (0-40%) ITS vehicle penetration rates, an effect that had not been observed in other scenarios. This is the result of DR optimising the routes taken by the ITS-vehicles and was examined in detail in Section 7.5 of this chapter.

  - Compared to ILC, the simplest adaptive ICA considered in this work, the combined effect of the Two-Step intersection management, ITAO, DR and CACC platooning, has decreased energy expenditure by 37% of the base energy consumption.

- The simulation results for the low speed limit (high traffic intensity) variant of the examined scenario are shown in Figure 7.23. The most significant

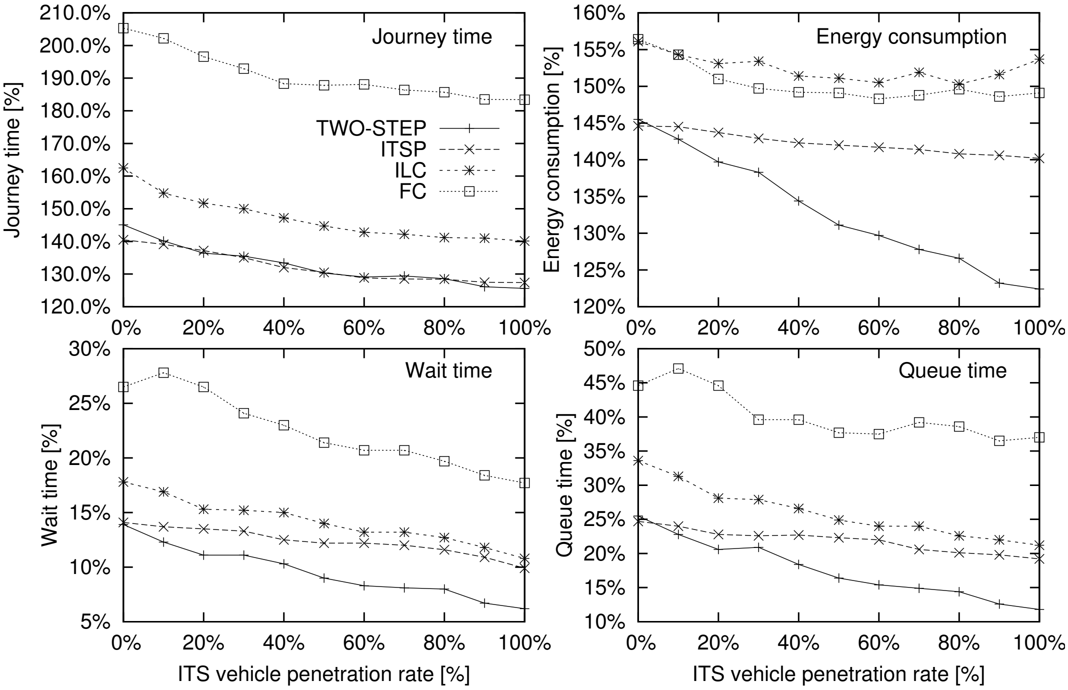


Figure 7.22: The performance of the investigated ICA in high intensity traffic flow conditions in the suburban speed variant of the Grid City scenario.

differences from the high speed version include:

- Smaller differences between journey times when using adaptive ICA are caused directly by the lower speed limit. Owing to the vehicles carrying less kinetic energy, the speed coefficient in the vehicular pressure equation is smaller, reducing the difference between the way ILC and pressure based algorithms work.
- An increase in energy consumption is observed in high ITS vehicle concentration rates when using the fixed cycles intersection control. Such an effect was observed and explained in previous scenarios, see Subsection 7.3.3, however this time it is more pronounced with FC than with ILC.

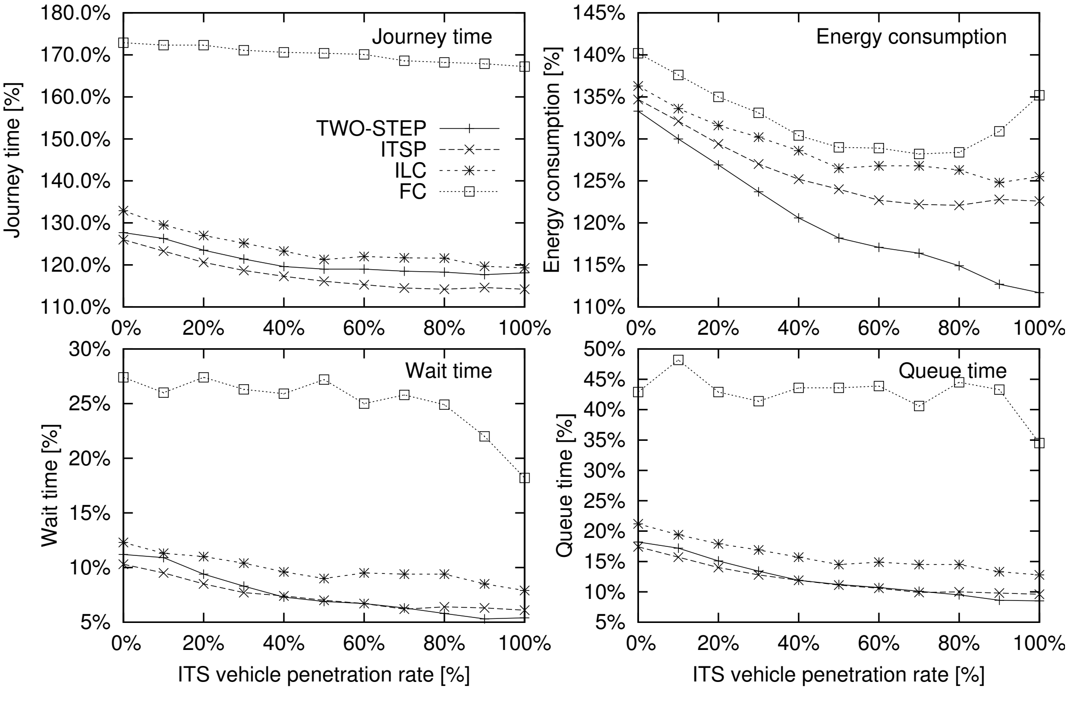


Figure 7.23: The performance of the investigated ICA in high intensity traffic flow conditions in the urban speed variant of the Grid City scenario.

7.10.4 Summary

This section presents a summary of the findings associated with the evaluation of the novel Two-Step traffic optimisation method, against ITSP, ILC and FC, using three different scenarios. Tables 7.7 and 7.8 summarise algorithm performance against ILC with 0% ITS-vehicle penetration rate. High traffic intensity is considered as it is more challenging and offers the most scope for improvement.

In low traffic intensities, with the exception of the Two-Step method which significantly reduced E_C, the observed differences in performance between the adaptive traffic control methods were small (see Subsection 7.10.1).

Table 7.7: Journey time T_J summary

	%ITS	Intersection control algorithm			
		FC	ILC	ITSP	Two-Step
Coventry	0	-71.1%	0.0%	**8.8%**	7.2%
	50	-57.5%	2.9%	**10.7%**	9.2%
	100	-60.7%	4.9%	**12.2%**	10.2%
Grid City	0	-26.3%	0.0%	**13.5%**	10.7%
	50	-15.6%	11.0%	19.7%	**19.8%**
	100	-12.8%	13.8%	21.6%	**22.7%**
Arterial	0	10.6%	0.0%	**53.8%**	47.1%
	50	25.6%	-9.1%	**62.7%**	58.7%
	100	28.7%	10.4%	**67.2%**	64.2%

Pressure based ICA (ITSP and Two-Step) outperformed the remaining intersection management methods. ITSP has shown to be slightly better than Two-Step in terms of T_J in two of the examined scenarios. Such outcome is expected assuming 0% ITS-vehicle concentration rates. The reason is that the Two-Step relies on MiSPS to provide prediction of future situation image that cannot be completely accurate, thus reducing the effectiveness of intersection management. Usually this is compensated by the ability to use IATO, however that requires the vehicles to be appropriately equipped.

Furthermore it was observed that higher ITS-vehicle concentration rates do not guarantee the Two-Step superiority over ITSP in terms of T_J. This is observed in the Arterial and Coventry scenarios, where the intersections are close to each other (see Section 7.4).

Table 7.8: Energy consumption E_C summary

	%ITS	Intersection control algorithm			
		FC	ILC	ITSP	Two-Step
Coventry	0	0.6%	0.0%	**8.0%**	7.7%
	50	14.3%	3.1%	12.6%	**16.2%**
	100	6.3%	5.5%	13.1%	**18.8%**
Grid City	0	-0.2%	0.0%	**7.4%**	6.8%
	50	4.5%	3.2%	9.0%	**16.0%**
	100	4.5%	1.6%	10.2%	**21.6%**
Arterial	0	**3.2%**	0.0%	-8.2%	-8.3%
	50	**7.5%**	-1.6%	-4.0%	-1.1%
	100	0.2%	-10.5%	-0.9%	**9.8%**

Table 7.8 summarises the investigated traffic management methods with respect to E_C.

Provided that sufficient ITS-vehicle penetration rate has been achieved, the novel Two-Step approach has shown to be superior in terms of E_C to the remaining methods. The sufficient ITS-vehicle penetration rate varies between scenarios and the traffic flow.

In congested traffic (Arterial Road scenario) the superior performance in terms of T_J is offset by the significantly increase in E_C observed in low ITS-vehicle concentration rates. The E_C is then driven down as the amount of ITS-vehicles increases. When demonstrating the benefits of their traffic management approaches authors in [8, 21, 22, 23, 35, 65, 148, 149, 150] did not measure the impact of their solution on energy consumption. It has been found in this work that in some of the cases reduction of journey times or delays can be offset by an increased energy consumption and CO_2 emissions.

7.11 Conclusions and findings summary

This chapter has evaluated the Cloud based Traffic Management System (CTMS) and its subcomponents using three distinct traffic network configurations. The components of the system were evaluated both independently and in combination with the others.

The following published findings were confirmed:

- CACC platooning:
  - Provides string stability, see [96] and Subsection 7.3.1.
  - Increases road throughput, see [96] and Subsection 7.3.2.
- The intersection approach optimisation mechanism has shown to improve T_J slightly and offers significant reduction in E_C, see [33, 40] and Section 7.4.
- Increasing speed limit increases road throughput, see [96] and Section 7.8.4.
- In congested traffic conditions the best performance is achieved by maximising the stage durations, see [123] and Subsection 7.10.2.

The following new findings were made:

- The novel Two-Step traffic management approach has shown to successfully combine the benefits of adaptive intersection control and intersection approach optimisation. In all investigated scenarios the new method achieved similar performance in terms of T_J to the ITSP, on which it is based on, while achieving much greater E_C savings. Combining IATO with an adaptive intersection management scheme yielded very good results. While IATO coupled with the FC intersection management method reduced E_C by 6.4% E_C^{base} from 156.4% E_C^{base} (0% ITS-vehicles) to 150% E_C^{base} (100% ITS-vehicles), using the Two-Step approach reduced E_C by 23.1% E_C^{base} from 145.5% E_C^{base} (0% ITS-vehicles) to 122.4% E_C^{base} (100% ITS-vehicles) in the same scenario (see

Section 7.4 and Subsection 7.10.3).

- The Two-Step method relies on the existence of ITS-vehicles to perform intersection approach optimisation and is therefore unsuitable to operate in 0% ITS-vehicle penetration rates. While in such conditions it is still usually superior to ILC and FC, the ITSP method is a better choice.
- The Two-Step method is beneficial in terms of E_C even for low ITS-vehicle penetration rates. In most of the investigated scenarios the Two-Step achieved E_C superiority over ITSP starting with 10% to 20% of ITS-vehicle penetration rates (see Figures 7.19, 7.20, 7.22 and 7.23).
- Situations where the vehicles enter the IATO range, while travelling significantly slower than the assumed speed limit, can cause a small number of vehicles (in each signalling stage) to assume suboptimal intersection approach decreasing T_J performance (see Section 7.4). Such situations occur in scenarios where intersections are so close to each other that when leaving one intersection, the vehicle enters IATO range of the next one without having achieved cruising speed.
- The manual vehicles can also benefit from the IATO if they are forced to follow an ITS-vehicle. In scenarios where there was no overtaking possible the IATO has shown to be most effective with ITS-vehicle penetration rates of 30% with very little improvement observed beyond that point (see Figure 7.12). In scenarios where overtaking is possible the non ITS-vehicles, unaware of the intersection approach optimisation, will overtake the ITS-vehicle optimising its approach. In such cases the IATO E_C benefits are achieved gradually, with the best performance being achieved when all interfering non ITS-vehicles are eliminated (see Section 7.4).
- The experiments have shown that using CACC greatly improves road throughput and journey times, however paired with an inappropriate intersection management

technique CACC can lead to significantly increased energy consumption. This phenomenon, described in Subsection 7.3.3, is caused by greater kinetic energy density in the CACC platoon compared to the ACC platoons. Use of the inappropriate intersection management method increases the probability of CACC platoon break-up by the traffic light resulting in increased energy losses in vehicles that were forced to stop.

- The Arterial Road scenario analysis has demonstrated that a fixed cycle intersection control can be the most appropriate approach in congested conditions. The adaptive schemes aim to optimise the traffic flow based on some criteria and identify which stage to activate. In congested traffic, where the vehicles are queuing on all sides of the intersection, the conditions for stage activation are similar on all approaches. In such conditions the throughput is maximised by using long signalling stages.
- It was observed in Subsection 7.10.2 that in congested conditions the spill-backs from downstream intersections can confuse the induction loop based ILC intersection management method. ILC assumes that all the queued vehicles will pass the intersection once the appropriate stage is activated for a sufficient amount of time (see Subsection 3.1.2 of Chapter 3). The vehicle counter associated with an induction loop is then reset. If the vehicles were unable to clear the intersection the amount of vehicles left queued is underestimated which leads to suboptimal intersection control.
- An interesting effect where in low ITS vehicle penetration rates smart vehicles are separated from the vehicle stream by following DR alternative route advice. Such filtered out ITS vehicles are much more likely to form platoons as the probability of a non-equipped vehicle interfering is lower. This effect supports the idea of creating dedicated lanes for ITS-vehicles, such as the CACC platoon lanes investigated in [41].

- The CTMS has shown to be tolerant to network delays expected to occur in metropolitan networks. The design of the novel Two-Step method accounts for processing delay and network latency. Large network delays will cause reduction in the advice horizon H_a and therefore slight degradation in traffic flow performance, however the IATO advice and the moment of traffic light change will remain timely.
- The CTMS has demonstrated an ability to configure itself automatically for the needs of the different traffic scenarios. The system requires static traffic information, such as the road layout and location of sensors to be provided. Using such information the system automatically configures the traffic management.

Chapter 8

Conclusions and Future Work

The previous chapter evaluated the performance of the developed Cloud based Traffic Management System using the purpose built traffic simulation tool.

This chapter concludes this work and highlights the novelties and their impacts on future ITS traffic management and platform development. A deployment plan for the developed system is outlined prior to discussing the limitations of the developed techniques and possible future research directions.

8.1 Conclusions

The research presented in this thesis has focused in two main areas. The first area addressed the issues of traffic management with the aims of reducing journey times and energy consumption in an urban environment. This was achieved by creating the Cloud based Traffic Management System (CTMS) that integrates novel traffic management methods, developed in this work, with existing Intelligent Transport Systems (ITS) techniques.

The second area of work addressed the issues of managing information obtained from distributed data sources and challenges of distributed processing. The traffic management performance was further improved using increased traffic situation awareness provided by distributed data management techniques in the novel ITS-Cloud platform. The ITS-Cloud endeavoured to increase flexibility, scalability and reliability of the novel urban traffic management system.

The developed traffic management methods were evaluated against two benchmarks, namely the fixed cycle and the adaptive intersection control algorithm exploiting traffic flow information form induction loops. By contrast to the literature the methods and algorithms were evaluated against both journey times and energy consumption for a fairer comparison. The benefits of the ITS-Cloud were also assessed against the additional burden associated with network communication.

The reported incompatibility between adaptive intersection control and intersection approach optimisation techniques has prompted the need for a new traffic management approach. The novel Two-Step traffic management scheme was created to address the aforementioned issue and is one of the fundamental contributions of this work. It exploits a traffic prediction mechanism combined, using the ITS-Cloud platform, with the ITS pressure (ITSP) adaptive intersection management algorithm to select the most appropriate signal timing. The delayed implementation of the timing decision enables the intersection controller to advise vehicles on the most appropriate intersection approach trajectory. The simulation studies have shown that the novel technique was successful in combining the benefits of two previously incompatible approaches resulting in significant and simultaneous reduction of journey times, energy consumption and CO_2 emissions. The method relies on the presence of ITS-vehicles appropriately equipped to receive the intersection approach instructions regarding the most appropriate speed profile to approach the intersection. The method was shown to be less effective than the underlying adaptive intersection control algorithm when there are no ITS-vehicles present. However the Two-Step method leads to significant gains even in low (20-30%) ITS-vehicle concentration rates.

The ITSP adaptive intersection control algorithm used in the Two-Step method was designed based on the concept of ITS pressure. It differs from the existing work by providing a new formulation for the vehicular pressure coefficient which exploits information on approaching vehicle speeds and locations, obtained from the ITS-sensors. Such information is then used to calculate the pressure and derive an appropriate signalling stage. Such method was evaluated independently and has shown to be successful in reducing both journey times and energy consumption compared to the benchmark methods.

The need for scalability, reliability and self configuration capabilities has prompted the development of a new processing platform meant for hosting traffic management system components. The platform was called ITS-Cloud and was developed with the purpose of hosting the CTMS and in future other ITS services. The novelty lies with the application of cloud computing mechanisms in the field of ITS with aim to provide the aforementioned benefits. The ITS-Cloud organises access to sensor data by constructing situation images of traffic in the intersection vicinity. A data integration technique is employed to construct such situation images using information obtained from different geographically distributed data sources such as lane sensors, V2I communication and information provided by the upstream intersections. Furthermore the specific requirements of ITS traffic management system prompted design and implementation of a new multi-dynamic service allocation method for the ITS-Cloud platform. Such a method has been implemented in the ITS-Cloud and combined the cloud and grid service allocation models. The multi-dynamic services benefit form increased reliability provided by the service duplication and relocation mechanisms of the ITS-Cloud.

A further contribution is the creation of a dedicated microscopic CTMS integrated traffic simulation tool. This new tool enabled extensive and in depth studies of different traffic networks with signalled intersections. The modelled road network components include roads, lanes, static signs, variable message signs and wireless communication infrastructure. Additionally it is capable of modelling behaviours such as adaptive cruise control (ACC), cooperative ACC (CACC), lane changing, dynamic routing and intersection approach optimisation. The probability of lane change, following distance and accepted safety margins are different for each vehicle and are part of the driver modelling in the simulator. The vehicle model used in the simulator was created and validated using experimentally obtained data. The simulator randomises the model parameters to simulate a range of different vehicles. Sensors, VMS and wireless communication nodes were interfaced with the ITS-Cloud interface which presented them as a cloud services. The microscopic modelling of traffic flow enabled simulating important ITS behaviours such as cooperative platooning or intersection approach optimisation. The nanoscopic aspect of vehicle modelling enabled energy consumption estimations.

Finally the developed methods and algorithms were integrated within the unique Cloud based Traffic Management System that was developed in the form of software services and deployed on the ITS-Cloud. This novel traffic management solution benefits form the scalability, self configuration capability, reliability and data management capabilities of the underlying ITS-Cloud platform. It comprises four intersection control algorithms (ICA) namely FC, ILC, ITSP and the novel Two-Step ICA. In addition to intersection control it is capable of coordinated traffic management in larger areas and provides dynamic routing advice.

The scalability and flexibility of the system was demonstrated in simulation by using the system to control traffic on different road networks, using different road, lane layouts and speed limits. In all of the cases, after being provided basic information about the road network, the CTMS automatically configured itself and provided the area with benefits of smart traffic management.

A unique aspect of this work included studying multiple ITS technologies together and evaluating their relative and combined impact on each other. The technologies studied included adaptive intersection management, intersection approach optimisation, cooperative platooning and dynamic routing. The individual component investigations have confirmed the observations made in the literature. The adaptive intersection control approaches excelled in reducing journey times and usually provided some energy savings as well. Cooperative platooning was shown to increase road throughput and platoon stability that contributed towards reducing journey times and energy consumption. The intersection approach optimisation has shown to provide significant energy savings and in some scenarios small decrease in journey times. Investigating the ITS components together has unveiled an issue caused by the lack of cooperation between cooperative platooning and some intersection management algorithms. A significant increase in energy consumption and CO_2 emissions was observed when a CACC platoon is broken by an untimely signalling stage change.

There was no proprietary or licensed software or libraries used in the implementation of CTMS, ITS-Cloud, the traffic simulator and supporting applications. All the components were implemented in *Java SE* and they can be deployed on any computer with the *Java runtime environment* installed.

The overall outcome of the work marks a significant milestone towards improving urban traffic management. The developed traffic management solutions addressed the issue by integrating existing traffic management methods with novel solutions. The platform on which the aforementioned solutions were deployed provided scalability, flexibility and reliability of the traffic management system as a whole.

8.2 Deployment plan

The CTMS was designed and implemented with future commercialisation and deployment in mind. The abstraction layer between the hardware and the traffic management system provided by the ITS-Cloud enables different kinds of traffic management hardware, such as lane sensors or traffic light controllers, to be integrated with the CTMS.

8.2.1 MIRA deployment

It is planned to hold the first real world tests of CTMS on the *innovITS Advance* city circuit on the MIRA premises. This state of the art test track was dedicated to testing and validating various ITS technologies. It is equipped with various forms of wireless communication infrastructure, traffic lights controllable through the network, variable message signs, and induction loops on some approaches. Even though there are no video sensing capabilities at this time, sensors referred to as *ITS sensors* in this work will be emulated by equipping each vehicle on the test track with a global navigation satellite system (GNSS) receiver and using it to track the vehicle's position. GNSS positioning will also be used to track the behaviour of the vehicles and collect data which will be used to assess the traffic management system's performance.

The Network Assisted Vehicle (NAV) was developed for use on the *innovITS Advance* test track (see Appendices F and G). Its primary purpose is ITS test automation and it is capable of receiving and following remotely received speed demands. It will be therefore used as what is referred to as an *ITS-vehicle* in this work to demonstrate the intersection approach optimisation capability of the CTMS.

After demonstrating the system in a closed and tightly controlled environment such as the MIRA proving ground, it is planned to deploy the system in a real city.

8.2.2 Coventry city deployment

Coventry city was chosen as an area for potential real life demonstration of the CTMS for multiple reasons. Close cooperation between Coventry City Council, MIRA Ltd. and Coventry University have led to the traffic data form the existing urban traffic management system to be made available for use in this project. CTMS can make use of pre-existing infrastructure in Coventry. Central Coventry is equipped with *NOW Wireless Mesh 4G* mesh network, which is currently mainly used to relay estimated bus arrival times that are displayed at the bus stops within the city. Induction loop sensors have been installed in multiple locations throughout the city and are being used with the SCOOT traffic management system. Such sensors are connected to the traffic management network in Coventry and simple software wrapper will need to be created to connect them to the CTMS as sensor services.

It is planned to equip public use vehicles such as buses, taxis and city council vehicles with a advisory system that would inform the drivers on how should they approach the intersection.

8.3 Limitations and future work

The CTMS in its current state is a complete and fully operational traffic management system, however there are several areas that can be either improved or extended. The future work and limitations are divided into four major sections related to the traffic simulation tool, the ITS-Cloud platform, the Cloud based Traffic Management System and urban traffic scenarios.

The current limitations and possible future improvement of the traffic simulator includes:

- **Vehicle model extension**
  A simplified vehicle model is currently used (see Section 6.3 of Chapter 6). Its design represents a trade-off between computational performance and the amount of detail simulated. It is planned to create a more detailed model to enable modelling of vehicle sub components with aim to capture effects such as different road surfaces, weather conditions or the impact of traffic management system on mechanical wear of the vehicle components.
- **Emission and fuel consumption model integration**
  The vehicle model used in the traffic simulator can only model the energy consumption using simplifying assumptions. It is planned to integrate fuel consumption and emission models to be able to model the amount and type (CO_2, NO_x etc.) of emission as well as the amount of fuel consumed.
- **Hybrid Electric Vehicle (HEV) power train model**
  Hybrid electric vehicles and other means of energy saving and recovery have become an important research area. Integrating a HEV power train model with the traffic simulation tool would greatly extend its application area. Benefiting from advance traffic information form CTMS a new regenerative braking strategies could be developed and new energy management techniques could be explored.
- **Macroscopic link model**
  The simulator aims to provide a universal, flexible and scalable platform for ITS traffic simulation. Using the ITS Cloud framework the simulation can be scaled to represent very large areas on a microscopic scale (see Section 6.1 of Chapter 6), however such level of detail is not always needed or justifiable. It is planned to improve the current implementation of Link-Model with a macroscopic link model that would be used to simulate long sections of road, such as motorways between ramps.

- **Road surface simulation**
  While the ITS traffic simulation tool does not aim to model events such as tyre slips and anti-lock braking systems, it does model energy consumption in the road network. Energy consumption varies depending on the road surface, weather conditions and elevation, all of which are not implemented in the current version of the tool. Implementing those features would increase the detail level of the simulation capabilities.
- **Advanced road definitions**
  Currently roads are defined as straight links between intersections. This limits the variety of scenarios that could be simulated. Implementing advanced road definitions that would enable roads of any shape to be simulated would broaden the application range of the simulation tool.
- **Component failure**
  In order to investigate the consequences of a traffic network component failure it is necessary to extend the simulator with a failure scripting capability.

The limitations of the ITS-Cloud platform and the areas for future improvement include:

- **Improved service allocation**
  The current service allocation method aims to balance the amount of services allocated on each resource. This approach is valid as long as it is assumed that all services consume similar amount of resources. In order to improve load balancing in the system mechanisms such as performance indexes [151] or auctioning [152] might be applied in future.
- **Decentralised cloud management**
  The ITS-Cloud framework uses a central service registry to provide service discovery and allocation capabilities. Such approach, while valid for relatively small scale systems such as the CTMS, introduces a single failure point to the

system and a potential performance bottleneck. In order to address that a decentralised service discovery system could be used in future evolution of the ITS-Cloud.

- **Improved messaging format**

  The ITS-Cloud relies on the *Java* serialization mechanism to exchange information between the services. While this approach is widely used in many networked *Java* applications it might be worth investigating alternative messaging formats, possibly using compression to reduce bandwidth requirements.

- **System security**

  The security in the developed system was provided by means of network isolation. In order to protect the system from malicious users more sophisticated security measures, such as communication encryption and user authentication, will need to be implemented before the system is deployed on a public network.

The future work in the area of to the traffic management methods includes:

- **Self tuning methods**

  In this research various tuning parameters have either been assumed or established experimentally. Each of those parameters should be investigated in detail and their impact on traffic management in various conditions examined. The use of self tuning and system identification methods would enable parameter adjustment to adapt the behaviour of traffic management to specific conditions, thereby maximising its performance for a wide range of conditions.

- **IATO improvement**

  The evaluation of the IATO has revealed a weakness in the conservative approach adopted. It can cause slight degradation of the traffic management performance when the vehicle is restricted to reducing its speed or keeping it constant when approaching an intersection (see Section 7.4 of Chapter 7). Further research is required to investigate acceleration profiles whilst approaching an intersection and the impact of other vehicles.

- **Examine different CACC approaches**

  Throughout the literature search different approaches to CACC were identified. All of them relied on wireless exchange of vehicle status information, however the way vehicles acted on such information differed. In this work one of such approaches, assuming that all the vehicles attempt to match the behaviour of the platoon leader, was implemented and investigated in this work. It is desirable to compare the remaining CACC approaches with the implemented one and investigate their effect on urban traffic management.

- **Extension of the MiSPS mechanism**

  The Micro Scale Prediction Service uses microscopic simulations to perform short term predictions of traffic situation in intersection area. An appropriate vehicle model is required to achieve good results. Currently only a car model is available in MiSPS. Whilst such vehicle model was sufficient for the scope of the simulations carried out in this work, real world deployment would warrant the implementation of a range of vehicle models within MiSPS

- **Further investigation of the Two-Step method**

  In the Two-Step traffic management approach, after the prediction of future traffic situation is obtained using MiSPS, the ITSP is used to generate the intersection management decision. While the evaluations have shown that the ITSP performs well it might be worth investigating applications of different adaptive traffic management methods for use with the Two-Step method.

- **Variable prediction horizon for Two-Step**

  In the current implementation of the Two-Step method the prediction horizon is determined during ICS configuration and remains constant thereafter. It is speculated that adopting variable length prediction horizon could lead to further improvement of the traffic management efficiency. In such approach several predictions would be made, each with different prediction horizon and the prediction horizon with the most potential for improving traffic flow would be selected.

The developed traffic management system and the simulator enable investigation of other aspects of road traffic. The scenarios that could be investigated using the developed software include:

- **ITS vehicle lanes**
  Some publications suggest using dedicated lanes for CACC/ITS vehicles and investigate their impact on the overall traffic performance [41]. Such a feature could be added to the simulator without significant effort to extend its capabilities and increase the variety of supported traffic scenarios.
- **Emergency vehicle modelling**
  Both the simulator and CTMS can be expanded with emergency vehicle modelling and handling. The main challenge in the simulator would be to implement the unique manner the traffic reacts to an emergency vehicle. The CTMS would attempt to control the traffic to minimise the time required for the emergency vehicle to arrive at the incident scene.
- **Incident response**
  Using the emergency vehicle modelling it would be possible to evaluate the impact of the CTMS has on the incident response time. It would be aimed to decrease the travel time of ambulances, fire services and the police as much as possible.

References

[1] Vehicle Licensing Statistics: 2012. *Department of Transport, Driver and Vehicle Licensing Agency*, 2013.

[2] John Pucher, Nisha Korattyswaropam, Neha Mittal, and Neenu Ittyerah. Urban transport crisis in India. *Transport Policy*, 12(3):185–198, 2005.

[3] Phil Goodwin. The economic costs of road traffic congestion. *University College London*, 2004.

[4] Glen Weisbrod, Don Vary, and George Treyz. Measuring Economic Costs of Urban Traffic Congestion to Business. *Transportation Research Record: Journal of the Transportation Research Board*, 1839, January 2003.

[5] R. Arnott, A. De Palma, and R. Lindsey. Does providing information to drivers reduce traffic congestion? *Transportation Research Part A: General*, 25(5):309–318, 1991.

[6] Lino Figueiredo, Isabel Jesus, Tenreiro Machado, and Jose Rui Ferreira. Towards the development of intelligent transportation systems. In *IEEE Intelligent Transportation Systems Conference*, pages 1206 – 1211, 2001.

[7] Victor Gradinescu, Cristian Gorgorin, Raluca Diaconescu, Valentin Cristea, and Liviu Iftode. Adaptive Traffic Lights Using Car-to-Car Communication. In *IEEE 65th Vehicular Technology Conference*, pages 21 – 25, 2007.

[8] W. Wen. A dynamic and automatic traffic light control expert system for solving the road congestion problem. *Expert Systems with Applications*, 34(4):2370–2381, May 2008.

[9] M. Venkatesh and V. Srinivas. Autonomous Traffic Control System Using Agent Based Technology. *International Journal of Advancements in Technology*, 2(3):438–445, 2011.

[10] Aleksandar Stevanovic, Jelka Stevanovic, and Cameron Kergaye. Comparative Evaluation of Traffic Signal Optimizations and Green Light Optimized Speed Advisory Systems. In *International Scientific Conference on Mobility and Transport*, 2013.

[11] Robert Oertel, Tobias Frankiewicz, Lars Schnieder, and Peter Wagner. Field Operational Test of a new Delay-Based Traffic Signal Control Using C2I Communication Technology Robert Oertel. In *International Scientific Conference on Mobility and Transport*, 2013.

[12] Vineet Kumar, Lan Lin, and Daniel Krajzewicz. iTETRIS: Adaptation of its technologies for large scale integrated simulation. In *71st IEEE Vehicular Technology Conference*, 2010.

[13] J. Bell and P. McMullen. Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, 18(1):41–48, January 2004.

[14] Lu Xiao-Yun, Pravin Varaiya, Roberto Horowitz, Dongyan Su, and Steven Shladover. A Novel Freeway Traffic Control with Variable Speed Limit and Coordinated Ramp Metering by. *Transportation Research Record: Journal of the Transportation Research Board*, pages 55–65, 2011.

[15] J.E. Naranjo, C. Gonzalez, J. Reviejo, R. Garcia, and T. De Pedro. Adaptive fuzzy control for inter-vehicle gap keeping. *Intelligent Transportation Systems, IEEE Transactions on*, 4 (3):132–142, September 2003.

[16] Arne Kesting, Martin Treiber, Martin Schönhof, and Dirk Helbing. Adaptive cruise control design for active congestion avoidance. *Transportation Research Part C: Emerging Technologies*, 16(6):668–683, December 2008.

[17] N.B. Hounsell, B.P. Shrestha, J. Piao, and M. McDonald. Review of urban traffic management and the impacts of new vehicle technologies. *Institution of Engineering & Technology, Intelligent Transport Systems*, 3(4):419–428, 2009.

[18] Kevin Collins and Gabriel-Miro Muntean. An Adaptive Vehicle Route Management Solution Enabled by Wireless Vehicular Networks. In *IEEE INFOCOM Workshops*, pages 1–5, September 2008.

[19] Markos Papageorgiou, Christina Diakaki, Vaya Dinopolou, Apostolos Kotsialos, and Yibing Wang. Review of road traffic control strategies. *Review of Road traffic control strategies, Proceedings of the IEEE*, 91(12):2043–2067, 2003.

[20] H.T. Shandiz, Mohsen Khosravi, and Mahsa Doaee. Intelligent Transport System Based on Genetic Algorithm. *World Applied Sciences Journal*, 6(7):908–913, 2009.

[21] Kurt Dresner and Peter Stone. Multiagent Traffic Management: A Reservation-Based Intersection Control Mechanism. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 530–537, 2004.

[22] Henry Liu, Jun-Seok Oh, and Will Recker. Adaptive signal control system with online performance measure for a single intersection. *Transportation Research Record: Journal of the Transportation Research Board*, pages 131–138, 2002.

[23] Seung-Bae Cools, Carlos Gershenson, and Bart D'Hooghe. Self-organizing traffic lights: A realistic simulation. *Advances in Applied Self-organizing Systems*, pages 41–50, 2008.

[24] Tessa Tielert, Moritz Killat, and Hannes Hartenstein. The impact of traffic-light-to-vehicle communication on fuel consumption and emissions. In *Internet of Things (IOT)*, pages 1 – 8, 2010.

[25] S. Halle and B. Chaibdraa. A collaborative driving system based on multiagent modelling and simulations. *Transportation Research Part C: Emerging Technologies*, 13(4):320–345, August 2005.

[26] Dirk Helbing and Amin Mazloumian. Operation Regimes and Slower-is-Faster-Effect in the Control of Traffic Intersections. *The European Physical Journal B*, 70(2), 2009.

[27] D.A. Roozemond. Using intelligent agents for urban traffic control systems. In *Proceedings of the international conference on artificial intelligence in transportation systems and science*, pages 69–79. Citeseer, 1999.

[28] Chi-Ying Liang and Huei Peng. Optimal adaptive cruise control with guaranteed string stability. *Vehicle system dynamics*, 31:313–330, 1999.

[29] Dave McKenney and Tony White. Distributed and adaptive traffic signal control within a realistic traffic simulation. *Engineering Applications of Artificial Intelligence*, 26(1):574–583, January 2013.

[30] Lars B. Cremean, Tully B. Foote, Jeremy H. Gillula, George H. Hines, Dmitriy Kogan, Kristopher L. Kriechbaum, Jeffrey C. Lamb, Jeremy Leibs, Laura Lindzey, Christopher E.

Rasmussen, Alexander D. Stewart, Joel W. Burdick, and Richard M. Murray. Alice: An information-rich autonomous vehicle for high-speed desert navigation. *Journal of Field Robotics*, 23(9):777–810, September 2006.

[31] A. Kelly, O. Amidi, and M. Happold. Toward Reliable Off Road Autonomous Vehicles Operating in Challenging Environments. *The International Journal of Robotics Research*, 25 (5):449–483, May 2006.

[32] The "SCOOT" Urban Traffic Control System, http://www.scoot-utc.com/documents/1_SCOOT-UTC.pdf.

[33] Raj Kishore Kamalanathsharma and Hesham Rakha. Agent-based modeling of Eco-Cooperative Adaptive Cruise Control systems in the vicinity of intersections. *15th International IEEE Conference on Intelligent Transportation Systems*, pages 840–845, September 2012.

[34] S. Lammer and D. Helbing. Self-Control of Traffic Lights and Vehicle Flows in Urban Road Networks. *Journal of Statistical Mechanics: Theory and Experiment*, page P04019, 2008.

[35] I. Porche, M. Sampath, R. Sengupta, Y.-L. Chen, and S. Lafortune. A decentralized scheme for real-time optimization of traffic signals. In *IEEE International Conference on Control Applications held together with IEEE International Symposium on Intelligent Control IEEE International Symposium on Computer-Aided Contro*, pages 582 – 589, 1996.

[36] J. Sewall, D. Wilkie, P. Merrell, and M. C. Lin. Continuum Traffic Simulation. *Computer Graphics Forum*, 29(2):439–448, June 2010.

[37] D.A. Roozemond and J.L.H. Rogier. Agent controlled traffic lights. In *European Symposium on Intelligent Techniques*, pages 77–82, 2000.

[38] Tobias Haupenthal, Thorsten Schöler, and Florian Weichenmeier. Agent-based Traffic Control – A Cyber-Physical System Approach. In *International Scientific Conference on Mobility and Transport*, 2013.

[39] Michel Ferreira, Ricardo Fernandes, Hugo Conceição, Wantanee Viriyasitavat, and Ozan K. Tonguz. Self-organized traffic control. In *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking - VANET '10*, pages 85–90, New York, New York, USA, 2010. ACM Press.

[40] K.J. Malakorn and B. Park. Assessment of mobility, energy, and environment impacts of IntelliDrive-based Cooperative Adaptive Cruise Control and Intelligent Traffic Signal control. In *IEEE International Symposium on Sustainable Systems and Technology*, pages 1–6, 2010.

[41] Bart van Arem, Cornelie J. G. van Driel, and Ruben Visser. The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):429–436, December 2006.

[42] Jean-Yves Potvin, Ying Xu, and Ilham Benyahia. Vehicle routing and scheduling with dynamic travel times. *Computers & Operations Research*, 33(4):1129–1137, April 2006.

[43] M. Dorigo and G. Di Caro. Ant colony optimization: a new meta-heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, pages 1470–1477, 1999.

[44] Marsh Products. The Basics of Loop Vehicle Detection http://www.marshproducts.com/pdf/Inductive%20Loop%20Write%20up.pdf.

[45] Carlos Sun and SG Ritchie. Individual vehicle speed estimation using single loop inductive waveforms. *Journal of Transportation Engineering*, 125(6):531–538, 1999.

[46] Benjamin Coifman. Estimating travel times and vehicle trajectories on freeways using dual loop detectors. *Transportation Research Part A: Policy and Practice*, 36(4):351–364, 2002.

[47] Z. Sun, G. Bebis, and R. Miller. On-road vehicle detection using optical sensors: a review. In *The 7th International IEEE Conference on Intelligent Transportation Systems*, pages 585 – 590, 2004.

[48] Cristiano Premebida, Goncalo Monteiro, Urbano Nunes, and Paulo Peixoto. A Lidar and Vision-based Approach for Pedestrian and Vehicle Detection and Tracking. In *2007 IEEE Intelligent Transportation Systems Conference*, pages 1044 – 1049, September 2007.

[49] M. Wiering. Multi-agent reinforcement learning for traffic light control. *http://igitur-archive.library.uu.nl/math/2007-0330-200425/wiering_00_multi.pdf*, 2000.

[50] Ioannis Papamichail and Katerina Kampitaki. Integrated ramp metering and variable speed limit control of motorway traffic flow. In *17th IFAC World Congress*, pages 14084–14089, 2008.

[51] Victor Firoiu and Marty Borden. A study of active queue management for congestion control. In *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1435 – 1444 vol.3, 2000.

[52] Hesham Rakha and Michel Van Aerde. Comparison of Simulation Modules of TRANSYT and INTEGRATION Models. *Transportation Research Record: Journal of the Transportation Research Board*, 1566(1):1–7, January 1996.

[53] Wilco Burghout, H.N. Koutsopoulos, and I. Andreasson. Hybrid mesoscopic-microscopic traffic simulation. *Transportation Research Record: Journal of the Transportation Research Board*, 1934(1):218–255, 2005.

[54] Emmanuel Bourrel, Rue Maurice Audin, and Vaulx-en-velin Cédex. Mixing Micro and Macro Representations of Traffic Flow: a Hybrid Model Based on the LWR Theory. *Transportation Research Record: Journal of the Transportation Research Board*, 1852(1):193–200, 2003.

[55] Daiheng Ni. 2DSIM: A Prototype of Nanoscopic Traffic Simulation. In *Intelligent Vehicles Symposium*, pages 47–52, 2003.

[56] A. Kotsialos, M. Papageorgiou, C. Diakaki, Y. Pavlis, and F. Middelham. Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool METANET. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):282–292, 2002.

[57] Rui Jiang, Mao-Bin Hu, Bin Jia, Ruili Wang, and Qing-Song Wu. Enhancing Highway Capacity By Homogenizing Traffic Flow. *Transportmetrica*, 4(1):51–61, January 2008.

[58] N.T. Ratrout. A comparative analysis of currently used microscopic and macroscopic traffic simulation software. *The Arabian Journal for Science and Engineering*, 34(1):121–133, 2009.

[59] Dirk Helbing and Ansgar Hennecke. MASTER: macroscopic traffic simulation based on a gas-kinetic, non-local traffic model. *Transportation Research Part B*, 35:183–211, 2001.

[60] W. L. Jin. Multicommodity kinematic wave simulation model for network traffic flow. *Transportation Research Record: Journal of the Transportation Research Board*, 1883(1):59–67, 2004.

[61] Yi-Sheng Huang and Ta-Hsiang Chung. Modeling and Analysis of Urban Traffic Lights Control Systems Using Timed CP-nets. *Journal of Information Science and Engineering*, 24 (3):875–890, 2008.

[62] Mariagrazia Dotoli and Maria Pia Fanti. An urban traffic network model via coloured timed Petri nets. *Control Engineering Practice*, 14(10):1213–1229, 2006.

[63] Kurt Jensen, Lars Michael Kristensen, and Lisa Wells. Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer*, 9(3-4):213–254, March 2007.

[64] U. C. Berkeley, F. Carlos, Randall Cayford, Wei-hua Lin, and Carlos F. Daganzo. The NETCELL simulation package : Technical description. In *Research Reports, California Partners for Advanced Transit and Highways (PATH), Institute of Transportation Studies (UCB), UC Berkeley*, 1997.

[65] J. de Gier, T.M. Garoni, and O. Rojas. Traffic flow on realistic road networks with adaptive traffic lights. *Journal of Statistical Mechanics: Theory and Experiment*, 4:P04008, 2011.

[66] Yoshihiro Hamada, Yusuke Nojima, Ken Ohara, and Hisao Ishibuchi. A Simulation Study of Route Selection with Inter-Vehicle Communication. Technical report, 2007.

[67] K. Nagel and M. Schreckenberg. A cellular automaton model for freeway traffic. *Journal de Physique I*, 2(12):2221–2229, 1992.

[68] S.C. Wong, W.T. Wong, and C.M. Leung. Group-based optimization of a time-dependent TRANSYT traffic model for area traffic control. *Transportation Research Part*, 36(4):291–312, May 2002.

[69] Loren Bloomberg and Jim Dale. Comparison of VISSIM and CORSIM traffic simulation models on a congested network. *Transportation Research Record: Journal of the Transportation Research Board*, 1727(1):52–60, 2000.

[70] Martin Fellendorf and Peter Vortisch. Microscopic Traffic Flow Simulator VISSIM. In Jaume Barceló, editor, *Fundamentals of Traffic Simulation*, volume 145 of *International Series in Operations Research & Management Science*, pages 63–93. Springer New York, New York, NY, 2010.

[71] Andrew Cunningham and Jeffery Archer. Modelling the combined motorway and urban traffic conditions in the Norrtull–Brunnsviken network using VISSIM. *Centre for Traffic Simulation Research (KTH) and Sweco*, 2003.

[72] Justice Appiah, Bhaven Naik, LR Rilett, and PEY Chen. Development of a State of the Art Traffic Microsimulation Model for Nebraska. *Nebraska Transportation Center*, 2011.

[73] Gabriel Gomes, Adolf May, and Roberto Horowitz. Congested freeway microsimulation model using VISSIM. *Transportation Research Record: Journal of the Transportation Research Board*, 1876:71–81, 2004.

[74] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz. SUMO - Simulation of Urban MObility. In *International Conference on Advances in System Simulation*, pages 63–68, 2011.

[75] Mordechai (Muki) Haklay and Patrick Weber. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing*, 7(4):12–18, October 2008.

[76] Michael Behrisch, Jakob Erdmann, and Daniel Krajzewicz. Adding intermodality to the microscopic simulation package SUMO. *German Aerospace Center, Institute of Transportation Systems*, 2010.

[77] Daniel Krajzewicz and Georg Hertkorn. An example of microscopic car models validation using the open source traffic simulation SUMO. In *14th European Simulation Symposium of Simulation in Industry*, 2002.

[78] M. Piorkowski, M. Raya, and A.L Lugo. TraNS: realistic joint traffic and network simulator for VANETs. *ACM SIGMOBILE Mobile Computing and Communications Review*, 12(1): 31–33, 2008.

[79] J. Gozálvez, S. Turksma, L. Lan, F. Cartolano, and D. Krajzewicz. iTETRIS: the Framework for Large-Scale Research on the Impact of Cooperative Wireless Vehicular Communications Systems in Traffic Efficiency. In *Information and Communications Technologies (ICT-MobileSummit)*, 2009.

[80] C.C. Chien and C.Y. Cheng. Autonomous intelligent cruise control using both front and back information for tight vehicle following maneuvers. In *American Control Conference*, volume 5, pages 3091 – 3095. American Autom Control Council, 1999.

[81] Shahab Sheikholeslam and Charles Desoer. Longitudinal control of a platoon of vehicles, I: Linear model. *PATH research report UCB-ITS-PRR-89-3*, 1989.

[82] C. Hatipoglu and U. Ozguner. Longitudinal headway control of autonomous vehicles. In *Proceedings of the 1996 IEEE International Conference on Control Applications*, pages 721 – 726, 1996.

[83] Michael Short, Michael Point, and H. Qiang. Safety and Reliability of Distributed Embedded Systems: Simulation of Vehicle Longitudinal Dynamics. Technical report, 2004.

[84] Yulin Ma, Xinping Yan, Qing Wu, and Rui Zhang. Research on Intelligent Vehicle platoon Driving Simulation Experiment System under the Coordination between Vehicle and Highway. *Journal of Computers*, 5(11):1767–1774, November 2010.

[85] W. Dunbar and R. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 4, pages 4631 – 4636, 2002.

[86] R. Behringer, S. Sundareswaran, B. Gregory, R. Elsley, B. Addison, W. Guthmiller, R. Daily, and D. Bevly. The DARPA grand challenge - development of an autonomous vehicle. In *Intelligent Vehicles Symposium*, pages 226 – 231. IEEE, 2004.

[87] R. Rajamani. *Vehicle Dynamics and Control*. Springer, 2006.

[88] D. Swaroop and J.K. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, 41(3):349–357, 1996.

[89] Gerrit Naus, Jeroen Ploeg, and Maarten Steinbuch. Towards on-the-road implementation of cooperative adaptive cruise control. In *Proceedings of the 16th World Congress & Exhibition on Intelligent Transport Systems and Services (ITS World)*, 2009.

[90] D. Yanakiev and I. Kanellakopoulos. Variable time headway for string stability of automated heavy-duty vehicles. In *34th IEEE Conference on Decision and Control*, volume 4, pages 4077 – 4081. Ieee, 1995.

[91] P. Ioannou and Z. Xu. Throttle and brake control systems for automatic vehicle following. *I V H S Journal*, 1(4):345–377, 1994.

[92] J. Navarro, F. Mars, and M.S. Young. Lateral control assistance in car driving: classification, review and future prospects. *IET Intelligent Transport Systems*, 5(3):207 – 220, 2011.

[93] W. J. Schakel, Bart Van Arem, and Bart Netten. Effects of cooperative adaptive cruise control on traffic flow stability. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 759 – 764, 2010.

[94] C.Y. Liang and Huei Peng. String stability analysis of adaptive cruise controlled vehicles. *JSME International Journal Series C*, 43(3):671–677, 2000.

[95] Levent Güvenç, Ilker Altay, Billin Güvenç, and Eray Bozkurt. Cooperative adaptive cruise control implementation of team mekar at the grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1062–1074, 2012.

[96] Pedro Fernandes, Urbano Nunes, and Senior Member. Platooning With IVC-Enabled Autonomous Vehicles: Strategies to Mitigate Communication Delays, Improve Safety and Traffic Flow. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):91–106, 2012.

[97] A. Goldsmith, S.S. Mahal, and J.K. Hedrick. Effects of communication delay on string stability in vehicle platoons. In *IEEE Intelligent Transportation Systems Conference*, pages 625 – 630. Ieee, 2001.

[98] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, and Others. Stanley: The robot that won the DARPA Grand Challenge. *Journal of field Robotics*, 23(9):661–692, 2007.

[99] Jonathan Bohren, Tully Foote, Jim Keller, Alex Kushleyev, Daniel Lee, Alex Stewart, Paul Vernaza, J. Derenick, J. Spletzer, and Brian Satterfield. Little Ben: The Ben Franklin Racing Team's Entry in the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25 (9):598–614, 2009.

[100] Andreas Birk, Holger Kenn, and Thomas Walle. On-board Control in the RoboCup Small Robots League. *Advanced Robotics*, 14(1):27–36, 2000.

[101] Peter Hidas. Modelling lane changing and merging in microscopic traffic simulation. *Transportation Research Part C: Emerging Technologies*, 10(5-6):351–371, October 2002.

[102] Arne Kesting, Martin Treiber, and Dirk Helbing. General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record: Journal of the Transportation Research Board*, pages 86–94, January 2007.

[103] R.G. Herrtwich and G. Nöcker. Cooperative driving: taking telematics to the next level. In *Traffic and Granular Flow*, number March, pages 271–280. 2003.

[104] Soufiene Djahel, Mazeiar Salehie, Irina Tal, and Pooyan Jamshidi. Adaptive traffic management for secure and efficient emergency services in smart cities. In *EEE International Conference on Pervasive Computing and Communications*, pages 340 – 343, 2013.

[105] S. Biswas, R. Tatchikou, and F. Dion. Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. *IEEE Communications Magazine*, 44(1):74–82, January 2006.

[106] A. Boukerche, H. Oliveira, E. Nakamura, and A. Loureiro. Vehicular Ad Hoc Networks: A New Challenge for Localization-Based Systems. *Computer Communications*, 31(12):2838–2849, July 2008.

[107] Wolfgang Kiess, J Rybicki, and Martin Mauve. On the nature of inter-vehicle communication. In *Communication in Distributed Systems (KiVS)*, pages 1 – 10, 2007.

[108] J.J. Blum, a. Eskandarian, and L.J. Hoffman. Challenges of Intervehicle Ad Hoc Networks. *IEEE Transactions on Intelligent Transportation Systems*, 5(4):347–351, December 2004.

[109] Meenakshi Bansal, Rachna Rajput, and Gaurav Gupta. Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations. *Mobile Adhoc Network (MANET) Working Group, IETF RFC 2501*, 1999.

[110] Ivan Tashev, Michael Seltzer, YC Ju, YY Wang, and Alex Acero. Commute UX: Voice enabled in-car infotainment system. In *Mobile HCI, Workshop on Speech in Mobile Pervasive Environments*, 2009.

[111] Mario Gerla and Leonard Kleinrock. Vehicular networks and the future of the mobile internet. *Computer Networks*, 55(2):457–469, February 2011.

[112] Guido Hiertz, Dee Denteneer, Lothar Stibor, Yunpeng Zang, X.P. Costa, and Bernhard Walke. The IEEE 802.11 universe. *IEEE Communications Magazine*, 48(1):62–70, January 2010.

[113] Daniel Jiang and Luca Delgrossi. IEEE 802.11p: Towards an International Standard for Wireless Access in Vehicular Environments. *VTC Spring 2008 - IEEE Vehicular Technology Conference*, pages 2036–2040, May 2008.

[114] William D Gropp, Dinesh K Kaushik, David E Keyes, and Barry F Smith. High-performance parallel implicit CFD. *Parallel Computing*, 27(4):337–362, March 2001.

[115] I. Foster and C. Kesselman. *The grid: blueprint for a new computing infrastructure.* 2004.

[116] J Shiers. The worldwide LHC computing grid (worldwide LCG). *Computer physics communications*, 177(1-2):219–233, 2007.

[117] Latha Srinivasan and Jem Treadwell Hp. An Overview of Service-oriented Architecture, Web Services and Grid Computing Service-Oriented Architecture. *HP Software Global Business Unit publication*, 2005.

[118] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud Computing and Grid Computing 360-Degree Compared. In *2008 Grid Computing Environments Workshop*, pages 1–10, November 2008.

[119] L.M. Vaquero, L. Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, 2008.

[120] G.V. McEvoy and B. Schulze. Using clouds to address grid limitations. In *Proceedings of the 6th international workshop on Middleware for grid computing*, 2008.

[121] Jeremy Geelan. Twenty-One Experts Define Cloud Computing. *Cloud Expo, online article: http://cloudcomputing.sys-con.com/node/612375*, 2009.

[122] Tomas Zelinka and M. Svitek. Adaptive communications solutions in complex transport telematics systems. In *12th WSEAS international conference on Communications*, pages 206–212. World Scientific and Engineering Academy and Society (WSEAS), 2008.

[123] B. DeSchutter. Optimal Traffic Light Control for a Single Intersection. In *Proceedings of the 1999 American Control Conference*, pages 2195 – 2199, 1999.

[124] Sadayuki Tsugawa. Issues and recent trends in vehicle safety communication systems. *International Association of Traffic and Safety Sciences*, 29(1):7–15, 2005.

[125] Seon Hak Kim, Oh Jung Kwon, Deoksu Hyon, Seung Ho Cheon, Jin Su Kim, Byeong Heon Kim, Sung Tack Hwang, Jun Seok Song, Man Taeck Hwang, and Byeong Soo Oh. Regenerative braking for fuel cell hybrid system with additional generator. *International Journal of Hydrogen Energy*, 38(20):8415–8421, May 2013.

[126] F. Semet and E. Taillard. Solving real-life vehicle routing problems efficiently using tabu search. *Annals of Operations research*, 41(4):469, 1993.

[127] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59:345–358, 1992.

[128] Paweł Jaworski, Tim Edwards, Jonathan Moore, and Keith Burnham. Cloud Computing Concept for Intelligent Transportation Systems. In *IEEE Intelligent Transportation Systems Conference*, pages 931 – 936, 2011.

[129] A. Iamnitchi and I. Foster. On fully decentralized resource discovery in grid environments. In *Grid Computing—GRID*, pages 51–62. Springer, 2001.

[130] Lizhe Wang and Gregor von Laszewski. Scientific cloud computing: Early definition and experience. In *10th IEEE International Conference on High Performance Computing and Communications*, pages 825 – 830, 2008.

[131] L. Opyrchal and A. Prakash. Efficient object serialization in Java. In *19th IEEE International Conference on Distributed Computing Systems*, pages 96 – 101, 1999.

[132] T. Imielinski and J.C. Navas. GPS-based geographic addressing, routing, and resource discovery. *Communications of the ACM*, 42(4):86–92, April 1999.

[133] Maria Kihl, Mihail Sichitiu, Ted Ekeroth, and Michael Rozenberg. Reliable geographical multicast routing in vehicular ad-hoc networks. *Wired/Wireless Internet Communications*, pages 315–325, 2007.

[134] Grady Booch, R Maksimchuk, and M Engle. *Object-oriented analysis and design with applications.* Pearson Education India, 2006.

[135] Booch Grady, James Rumbaugh, and Ivar Jacobson. *The unified modeling language user guide.* Addison Wesley, 2005.

[136] M. Bauer and M Tomizuka. Fuzzy Logic Traction Controllers and Their Effect on Longitudinal Vehicle Platoon Systems. *Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility*, 25(4), 1996.

[137] Rahmi Akcelik and Mark Besley. Operating cost, fuel consumption, and emission models in aaSIDRA and aaMOTION. In *25th Conference of Australian Institutes of Transport Research*, 2003.

[138] Tim Edwards, Jonathan Moore, Maria Loukadaki, and Pawel Jaworski. A Network Assisted Vehicle for ADAS and ITS testing. In *IEEE Intelligent Transportation Systems Conference*, pages 681 – 685, 2011.

[139] Tomer Toledo, Charisma Choudhury, and Moshe Ben-Akiva. Lane-Changing Model with Explicit Target Lane Choice. *Transportation Research Record*, 1934(1):157–165, January 2005.

[140] M. Rickert and K. Nagel. Two lane traffic simulations using cellular automata. *Physica A: Statistical Mechanics and its Applications*, 231(4):534—-550, 1996.

[141] Tomer Toledo, Haris Koutsopoulos, and Moshe Ben-Akiva. Modeling Integrated Lane-Changing Behavior. *Transportation Research Record: Journal of the Transportation Research Board*, 1857:30–38, January 2003.

[142] Charles Macadam. Understanding and modeling the human driver. *Vehicle System Dynamics*, 40(734):101–134, 2003.

[143] Fan Bai, Daniel D. Stancil, and Hariharan Krishnan. Toward understanding characteristics of dedicated short range communications (DSRC) from a perspective of vehicular network engineers. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking - MobiCom '10*, pages 329–340, New York, USA, 2010. ACM Press.

[144] Yusuke Tanaka, Yuu Nakajima, Hiromitsu Hattori, and Toru Ishida. A driver modeling methodology using hypothetical reasoning for multiagent traffic simulation. In *Agent Computing and Multi-Agent Systems*, pages 278–287. Springer, 2009.

[145] Steffi Klinge and Richard H. Middleton. Time headway requirements for string stability of homogeneous linear unidirectionally connected systems. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC)*, pages 1992 – 1997, December 2009.

[146] Stephen Hemminger. Network Emulation with NetEm. Technical Report April, 2005.

[147] www.ping.ms - Free online ping test and traceroute service with multiple locations worldwide.

[148] M.B. Trabia, M.S. Kaseko, and Murali Ande. A two-stage fuzzy logic controller for traffic signals. *Transportation Research Part C: Emerging Technologies*, 7(6):353–367, 1999.

[149] Y.S. Murat and Ergun Gedizlioglu. A fuzzy logic multi-phased signal control model for isolated junctions. *Transportation Research Part C: Emerging Technologies*, 13(1):19–36, February 2005.

[150] Marco Wiering and J Veenen. Intelligent traffic light control. *Utrecht University: Information and Computing Sciences*, 2004.

[151] Rajkumar Buyya, CS Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *10th IEEE International Conference on High Performance Computing and Communications*, pages 5–13, 2008.

[152] Wei-Yu Lin, Guan-Yu Lin, and Hung-Yu Wei. Dynamic Auction Mechanism for Cloud Resource Allocation. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 591 – 592, 2010.

Appendices

Appendix A

Traffic simulator application

This appendix describes the functions and the interface of the traffic simulator application described in Chapter 6.

A.1 Scenario configuration

Before running a simulation it is necessary to configure the simulation scenario. The simulation scenario consists of the road network definition and information on how and where vehicles should be injected. A simple scripting format was introduced to enable text based configuration of the simulator. Every line of the configuration file contains a declaration of a simulation entity (see Section 6.2 of Chapter 6). Each entity has a name which must be unique in the scope of the simulation and a list of parameters that define its behaviour and relationship with other entities. There are following entity definitions available:

- **INTERSECTION** $< name >< x, y, z >< type >< parameters >$

  This command declares an intersection with a specified name, location, type and parameters:

  - $< x, y, z >$ - the first parameter is the set coordinates of the intersection in simulated space. The height z is optional and will be set to zero if omitted.

- $< type >$ - the second parameter defined the type of the intersection, currently two types are supported. *T1* is the generic intersection with up to four roads. *Link* defines a terminus for a Link-Model (see Section 6.8 of Chapter 6).
- $< parameters >$ - the last parameter is a list of semicolon separated sub-parameters that define the inner working of the intersection. The sub-parameters are optional and can be specified in any order.
  - $< node >= Lx, Ry$ - the first sub-parameter defines the amount of left turn lanes (e.g. L2 for 2 left turns) and right turn lanes for the appropriate node. A T1 intersection has 4 nodes: A1, A2, B1, B2. In the intersection context nodes are points where roads can be connected (see Section 6.5 of Chapter 6). This parameter can be specified for all four nodes of a *T1* type intersection. Those parameters are optional. If not specified the default values of one left and one right turn will be used.
  - $< join >$ - the second sub-parameter defines if the turning lanes can be shared with ahead lanes.

An example definition of an intersection is as follows:

```
INTERSECTION is1 50,50 T1 A1=L1,R1;A2=L1,R2;B1=L1,R1;B2=L1,R1;join
```

This defines a *T1* type intersection called *is1* located at point $x = 50$ $y = 50$ $z = 0$ in the simulated area. There is one left and one right turn from nodes $A1$, $B1$, and $B2$ and one left and two right turns from node $A2$. Joining of turning and ahead lanes is permitted. The amount of ahead lanes depends on the total amount of lanes of the road attached to the node. For example attaching 2 lane (in one direction) road to node $A1$ will result in the left lane allowing vehicle to turn left ($B1$ is the left turn from $A1$, see Section 6.5 of Chapter 6) and go straight (to the road linked to $A2$).

- **ROAD** $< name >< startIS : Node >< endIS : Node >< lanesUp, lanesDown >< speedLimit >$

  This command defines a road between two intersections. The first two parameters specify between which intersections should the road be created and which intersection nodes should be used. The third parameter allows specifying the amount of lanes in each direction. The fourth parameter defines the default speed limit on this road.
- **CAR** $< name >< laneID >< startPosition >< initialSpeed >$

  This command creates a vehicle in a specified position and sets its initial speed. Specifying a name starting with a capital letter results in creation of an ITS-vehicle. Otherwise a normal/manual vehicle is created (see Section 6.3 of Chapter 6).
- **RTPLAN** $< name >< listofintersections >$

  A route plan is a list of intersections that the vehicle should visit on its journey.
- **SPRATE** $< name >< smoothing >< time1, rate1 >$ $[time2, rate2]$[...] The injection rate plan used with vehicle spawners defines how vehicle injection rate varies with time. If smoothing is enabled the transitions between set points will be done gradually. The list of parameters consists of pairs which define vehicle injection rates and the moments of time when said rates should be used. If smoothing option is used the injection rate will linearly change in between the specified times. Without smoothing option the old injection rate is replaced with the new one once the specified time passes. Each plan can be used with multiple spawners.
- **SPAWNER** $< name >< laneId >< location >< startingSpeed >< clearDistance >< spawnRate >< \%ITS >< routePlan >$

This command creates a vehicle spawner that injects vehicles into the simulated area. The parameters are as follows:

  - $< laneId >$ - The first parameter specifies the lane on which the spawner should be placed.
  - *location* - The second parameter is the spawner location relative to the beginning of the lane (0 means the beginning of the lane and 100 is the end)
  - $< startingSpeed >$ - The third parameter is the initial speed of injected vehicles.
  - $< clearDistance >$ - The fourth parameter specifies the how far the previously created vehicle has to move away from the spawner to allow injection of following vehicle.
  - $< spawnRate >$ - This parameter specifies the vehicle injection rate. A name of a previously defined variable injection plan (SPRATE) can be provided here. alternatively a fixed injection interval can be specified.
  - $< \%ITS >$ - This parameter determines the proportions between normal/manual and ITS vehicles created by this spawner. Setting this to 100 will result in only ITS vehicles being created. Value of 0 will cause the spawner to create only normal vehicles.
  - $< routePlan >$ - The final parameter is optional and allows user to specify the path (RTPLAN) the created vehicles should follow. Normal/manual vehicles injected by the associated vehicle spawner will follow this route and ITS vehicles will rely on the DR mechanism (see Section 3.6 of Chapter 3) to reach their destination.

- **ILOOP** $< name >< laneId >< location >$

  This command places an induction loop at the specified location.

- **ITSS** $< name >< laneId >< location >$

Results in an ITS-sensor being created at the specified location.

- **VMS** $< name >< laneId >< location > [speedLimit]$

  Places a VMS at the desired location. Optionally an initial speed limit can be defined.

- **ALGORITHM** $< algotihmName >$

  Defines which intersection management method should be used (see Chapter 3). Available choices include:

  - *fc* - the fixed cycle based algorithm.
  - *ilc* - the induction loop count based intersection control algorithm.
  - *itsp* - the ITS pressure adaptive intersection management scheme.
  - *2step* - the Two-Step novel intersection management method.

- **REGISTRY** $< host : port >$

  Specifies the location of the ITS-Cloud SDS (see Section 4.4 of Chapter 4)..

- **AREA** $< name >< x, y >$

  Defines the name for the area simulated in this scenario. This field is used by ITS-Cloud to link areas using Link-Model.

An example of a single cross-type intersection is given below. The roads can only be defined between intersections therefore it is necessary to define additional intersections that define the boundaries of simulated road network. An intersection with only one road attached to it is referred to as dummy intersection in the simulator.

```
ALGORITHM ITSP
AREA Coventry 9834,5213
REGISTRY ServerName:1234


INTERSECTION middle T1 500,500 T1 join
INTERSECTION north T1 500,500 T1 join
INTERSECTION south T1 500,500 T1 join
INTERSECTION east link 500,500 LM:RemoteAreaName
INTERSECTION west T1 500,500 join

ROAD rNorth north:A1 middle:A2 2,2 15
ROAD rSouth south:A2 middle:A1 2,2 15
ROAD rEast east:B1 middle:B2 2,2 15
ROAD rwest west:B2 middle:B1 2,2 15

RTPLAN routePlanEast middle east

SPRATE eastSpawnRate smooth 1,8000 1200,8000 2400,2500 5400,8000
    7200,8000

SPAWNER spEast rEast_to_middle_0 1 30 10 eastSpawnRate 50 routePlanEast
```

Listing A.1: Example simulation configuration

A.2 Modes of operation

The traffic simulator can operate in two modes. The graphical user interface (GUI) mode enables the user to visualise the road network and location of the vehicles. The batch mode is designed to automatically run series of simulations without user supervision.

A.3 Batch mode

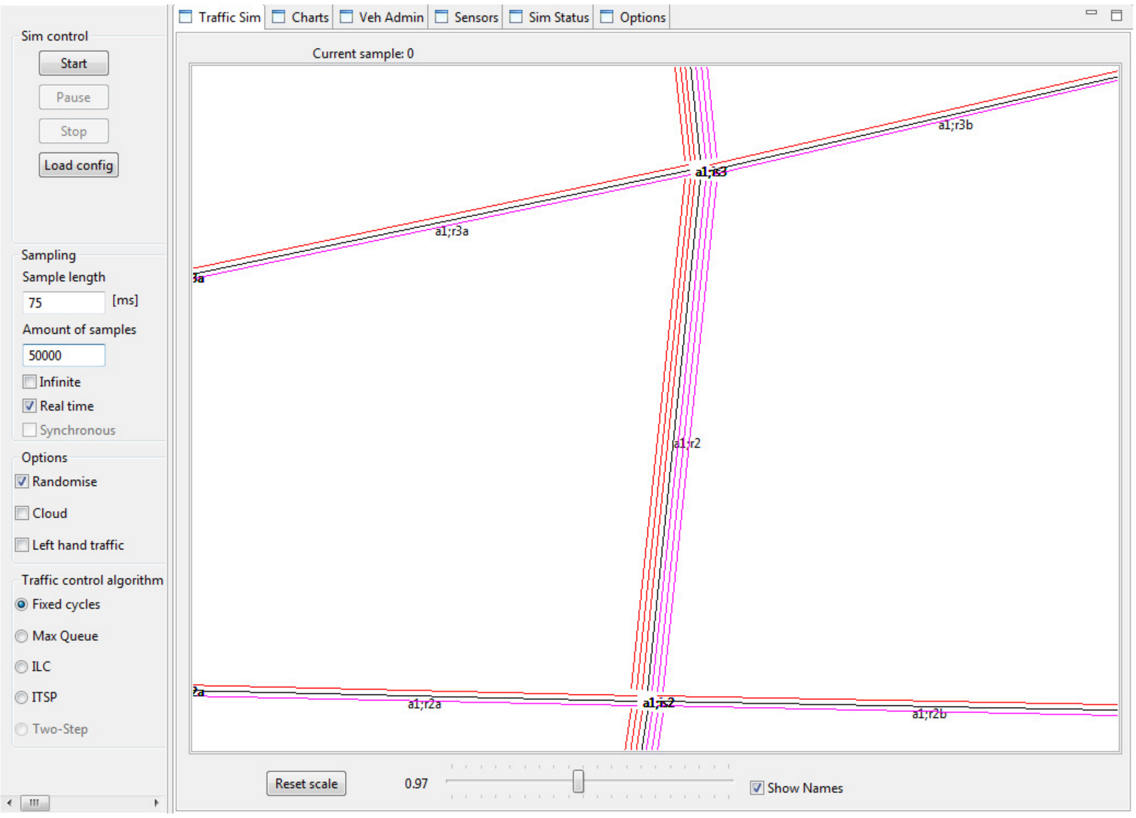
In order to run a batch mode simulation it is required to place all the desired scenario configuration files into a directory and invoke the simulation software as follows:

```
java −jar CTMS−SIM.jar −C <configDirectory> −r <samplingRate> −s
    <amountOfSamples>
```

The software will run the specified scenarios in order and save all the results in a subdirectory.

A.4 Graphical user interface mode

The GUI mode was meant for demonstrations and visualisation purposes. In GUI mode the simulation is configured and run using the application's graphical user interface.


Traffic Sim
Charts
Veh Admin
Sensors
Sim Status
Options
Sim control
Start
Pause
Stop
Load config
a1;r3b
a1;is3
a1;r3a
3a
a1;r2
2a
a1;r2a
a1;is2
a1;r2b
Reset scale
0.97

Current sample: 0

Sampling
Sample length 75 [ms]
Amount of samples 50000
☐ Infinite
☑ Real time
☐ Synchronous

Options
☑ Randomise
☐ Cloud
☐ Left hand traffic

Traffic control algorithm
● Fixed cycles
○ Max Queue
○ ILC
○ ITSP
○ Two-Step

☑ Show Names

Figure A.1: Main traffic stimulation view

Figure A.1 presents the main window of the traffic simulator. The rights side is occupied by the control panel, which is always visible regardless of the currently selected view. It allows users to start, stop and pause the simulation as well as provides means of loading different traffic scenario configuration files.

The sampling section enables users to specify sample lengths and the amount of samples to be simulated. It is possible to run the simulator for a undetermined amount of samples by checking the infinite check box. In such case the simulation will continue until explicitly stopped by the user. If real time option is chosen the time flow in the simulation is governed by the computer's real time clock. Real time option has to be enabled in order to perform asynchronous distributed simulations or when any component using ITS-Cloud platform is in use. When real time remains unchecked the simulation will execute as fast as the host processing power allows enabling simulating hours of traffic in few minutes.

Synchronous mode is only meaningful if the simulator is linked with other simulator instances through ITS-Cloud. It enables event synchronisation throughout the simulated realm which frees the components from having to use real time clock as the time base. The simulation executes as fast as the slowest component in the network. Event synchronisation introduces significant overheads in the network, which in case of large simulation scenarios can make synchronous mode slower than real time clock triggered asynchronous mode.

The options section gives users the possibility to enable or disable entity randomisation, enable ITS-Cloud platform and switch between left and right hand traffic. Enabling randomisation will cause parameters of various entities to be randomised to create a varied and realistic traffic scenario. Disabling randomisation makes all vehicles identical.

The final section of the control panel is the selection of the traffic control algorithm.

A.4.1 Traffic simulation view

The traffic simulation view is the basic default view in the traffic simulator application. It shows the road layout and current traffic situation in the simulated road network fragment.

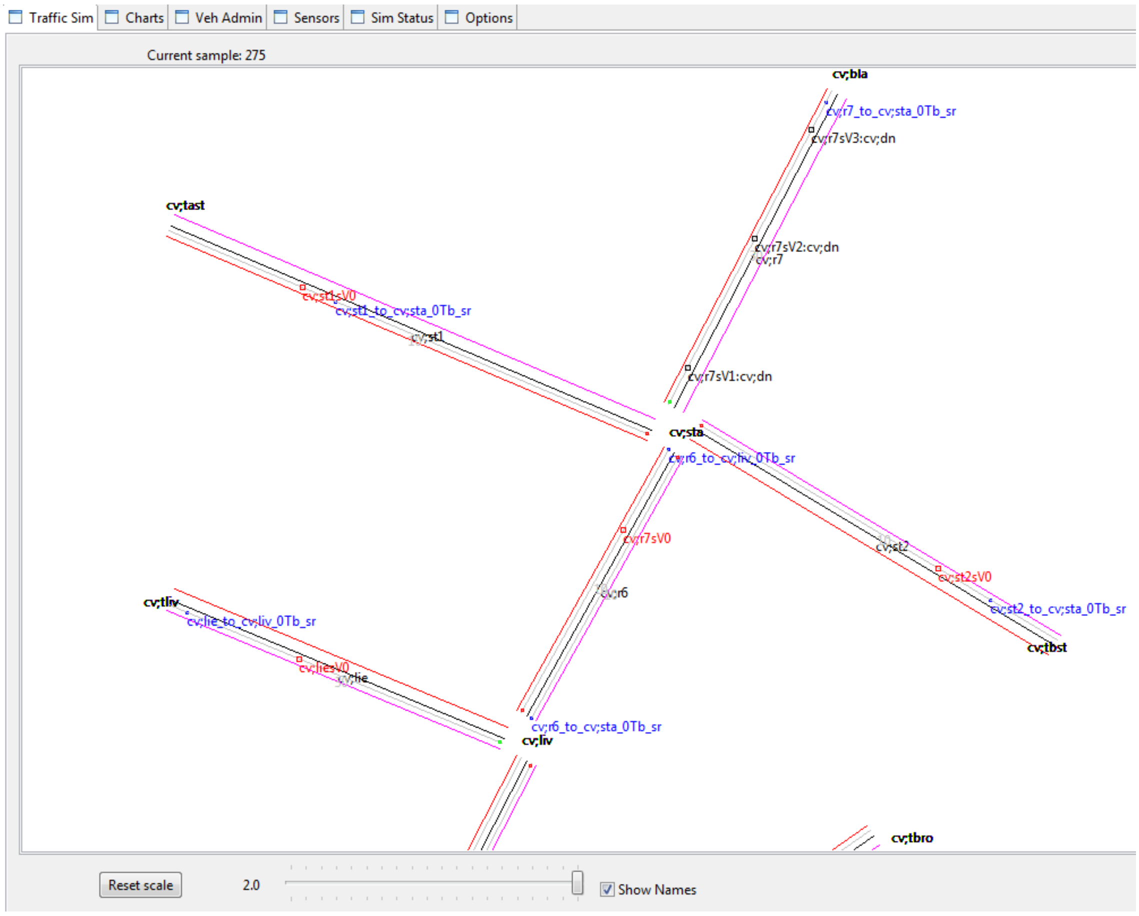

☐ Traffic Sim
☐ Charts
☐ Veh Admin
☐ Sensors
☐ Sim Status
☐ Options
Current sample: 275
cv;bla
cv;r7_to_cv;sta_0Tb_sr
cv;r7sV3:cv;dn
cv;tast
cv;r7sV2:cv;dn
cv;r7
cv;st1sV0
cv;st1_to_cv;sta_0Tb_sr
cv;st1
cv;r7sV1:cv;dn
cv;sta
cv;r6_to_cv;liv_0Tb_sr
cv;r7sV0
cv;st2
cv;st2sV0
cv;r6
cv;tliv
cv;lie_to_cv;liv_0Tb_sr
cv;st2_to_cv;sta_0Tb_sr
cv;tbst
cv;liesV0
cv;lie
cv;r6_to_cv;sta_0Tb_sr
cv;liv
cv;tbro
Reset scale
2.0
☑ Show Names

Figure A.2: Traffic simulation view

Figure A.2 illustrates an example of the traffic simulation view. The view is equipped with sample counter informing the user of the elapsed simulation time and zoom tools enabling viewing the simulated area in different zoom levels. An option to disable display of entity names can be used to make the view clearer in situations with many simulated vehicles and a dense road network. There are following views available:

A.4.2 Traffic simulation charts view

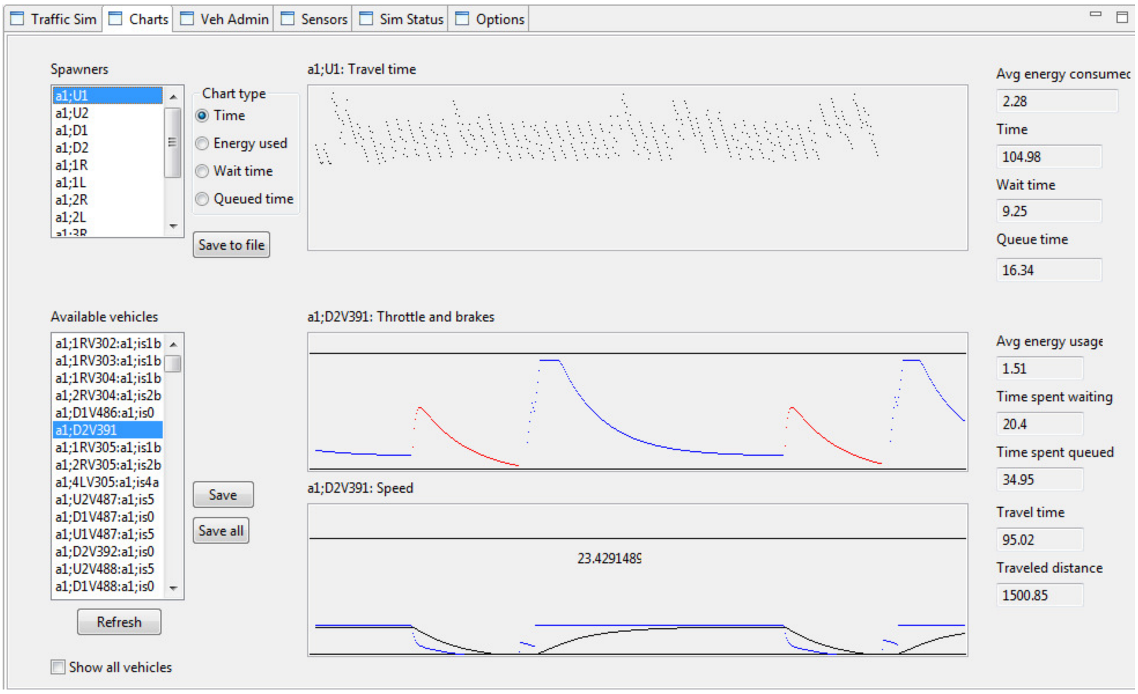

Traffic Sim
Charts
Veh Admin
Sensors
Sim Status
Options
Spawners
a1;U1
a1;U2
a1;D1
a1;D2
a1;1R
a1;1L
a1;2R
a1;2L
Save to file
a1;U1: Travel time
Available vehicles
a1;1RV302:a1;is1b
a1;1RV303:a1;is1b
a1;1RV304:a1;is1b
a1;2RV304:a1;is2b
a1;D1V486:a1;is0
a1;D2V391
a1;1RV305:a1;is1b
a1;2RV305:a1;is2b
a1;4LV305:a1;is4a
a1;U2V487:a1;is5
a1;D1V487:a1;is0
a1;U1V487:a1;is5
a1;D2V392:a1;is0
a1;U2V488:a1;is5
a1;D1V488:a1;is0
Refresh
Save
Save all
a1;D2V391: Throttle and brakes
a1;D2V391: Speed
23.4291489

Chart type
● Time
○ Energy used
○ Wait time
○ Queued time

☐ Show all vehicles

Avg energy consumec: 2.28
Time: 104.98
Wait time: 9.25
Queue time: 16.34

Avg energy usage: 1.51
Time spent waiting: 20.4
Time spent queued: 34.95
Travel time: 95.02
Traveled distance: 1500.85

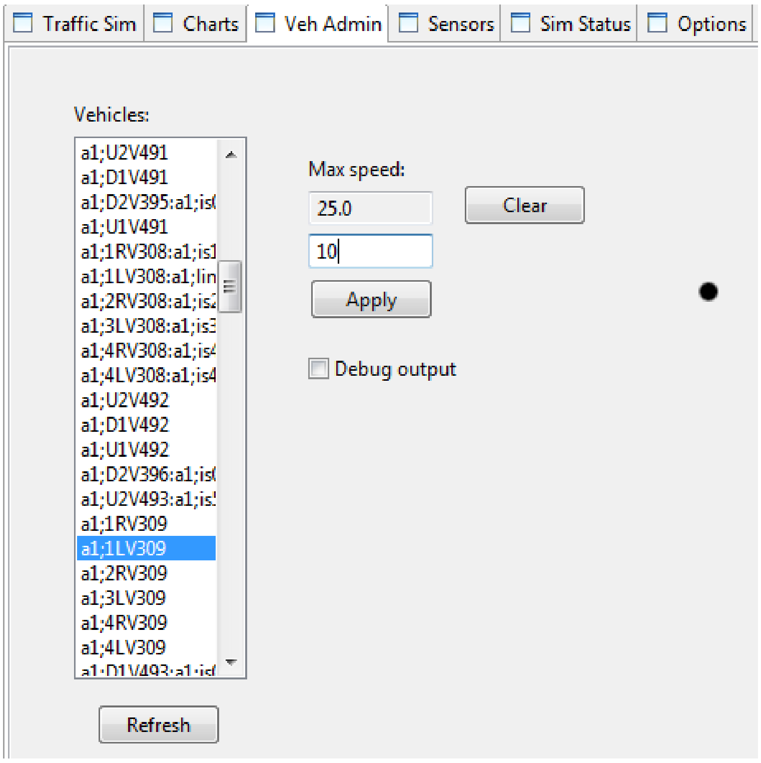
Figure A.3: Traffic simulation - charts

The charts view enables users to investigate current status of a selected vehicle and whole routes as shown in Figure A.3. The top chart allows users to view statistics of all vehicles injected by a spawner selectable from the list in the upper left corner of the application window.

The two lower charts display the history of throttle and brake usage as well as the current and desired speed of a vehicle selected from the list in lower left corner of the application window.

A.4.3 Vehicle administration view

Figure A.4 shows the vehicle administration view of the simulator. Such view is meant for debugging purposes and allows the user to override the speed limit for any vehicle in the simulation. This view can be used to simulate incidents or load blocks by stopping a vehicle in a desired location.



a1;U2V491
a1;D1V491
a1;D2V395:a1;is(
a1;U1V491
a1;1RV308:a1;is1
a1;1LV308:a1;lin
a1;2RV308:a1;is2
a1;3LV308:a1;is3
a1;4RV308:a1;is4
a1;4LV308:a1;is4
a1;U2V492
a1;D1V492
a1;U1V492
a1;D2V396:a1;is(
a1;U2V493:a1;is!
a1;1RV309
a1;1LV309
a1;2RV309
a1;3LV309
a1;4RV309
a1;4LV309
a1;D1V493:a1;is(

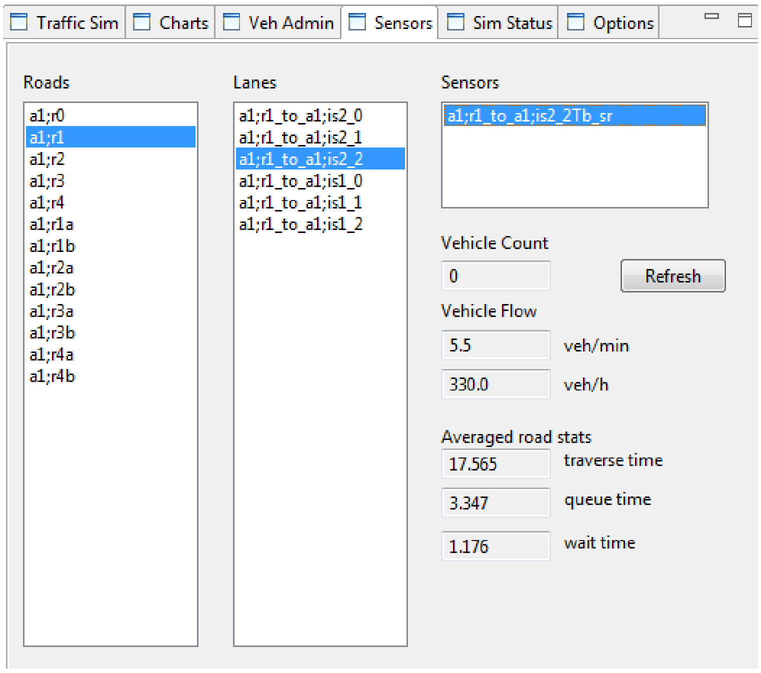
Max speed:
25.0
10

☐ Debug output

Figure A.4: Vehicle Administration

A.4.4 Sensor view

The view presented in Figure A.5 allows the user to obtain the current readings of any sensor in the simulation. The user selects a road from the left most list, then a lane from the middle list and finally an individual sensor from the right most list. The sensor reading are then displayed on the right and include:



Roads

a1;r0
a1;r1
a1;r2
a1;r3
a1;r4
a1;r1a
a1;r1b
a1;r2a
a1;r2b
a1;r3a
a1;r3b
a1;r4a
a1;r4b

Lanes

a1;r1_to_a1;is2_0
a1;r1_to_a1;is2_1
a1;r1_to_a1;is2_2
a1;r1_to_a1;is1_0
a1;r1_to_a1;is1_1
a1;r1_to_a1;is1_2

Sensors

a1;r1_to_a1;is2_2Tb_sr

Vehicle Count

0

Vehicle Flow

5.5 veh/min
330.0 veh/h

Averaged road stats

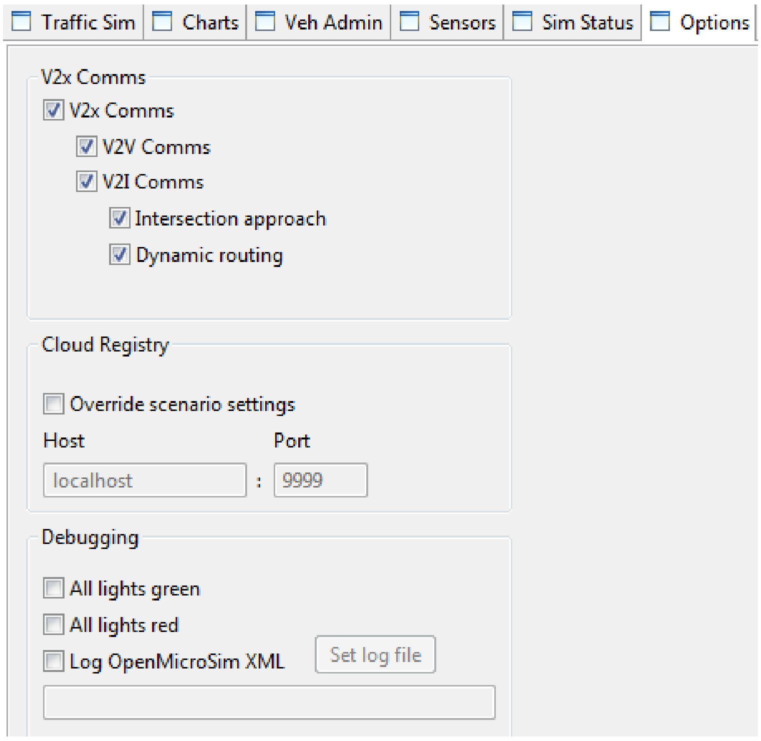
17.565 traverse time
3.347 queue time
1.176 wait time

Figure A.5: Traffic simulation sensors

- Vehicle count - the current amount of vehicles seen by an ITS-sensor or the amount of vehicles counted by an induction loop since the last counter reset.
- Vehicle flow - the average vehicle flow, averaged form the last 5 minutes.
- Averaged road statistics - The statistics for the road occupied by the sensor. The average the time vehicles spent waiting and queued on this road section as well as the average traversal time are available.

A.4.5 Options view

The options view presented in Figure A.6 allows the user to further customise the simulation environment. It is possible to disable various forms of wireless communications and subsystems. Disabling V2V communications will result in CACC platooning being unavailable. Disabling V2I communication will prohibit the simulated vehicles form benefiting from IATO and dynamic routing. Those two



V2x Comms

☑ V2x Comms
  ☑ V2V Comms
  ☑ V2I Comms
    ☑ Intersection approach
    ☑ Dynamic routing

Cloud Registry

☐ Override scenario settings

Host: localhost
Port: 9999

Debugging

☐ All lights green
☐ All lights red
☐ Log OpenMicroSim XML

Figure A.6: Traffic simulation options

options are configurable separately as well.

The next option allows the user to override the ITS-Cloud registry setting loaded form the scenario definition file.

Debugging options allow the user to override traffic light settings and force all to green or red phase. The simulator can also log all the events in an *OpenMicroSim* format.

Appendix B

ITS-Cloud software

The CTMS and ITS-Cloud have been packaged into Java Archives (jar files) for easy deployment. In order to obtain a functional ITS-Cloud platform it is necessary to create an instance of the Service Discovery System (SDS, see Section 4.4 of Chapter 4) and at least one resource (see Section 4.2 of Chapter 4). ITS-Cloud components can be created on any software platform with *Java SE* platform installed.

B.1 Deployment of ITS-Cloud registry

Every CTMS/ITS-Cloud deployment requires an instance of the Service Discovery System (SDS) to be present . The SDS can be deployed as follows:

```
java −jar CTMS.jar −R <port>
```

Listing B.1: Creation of SDS

It is configured to listen for requests on the defined port number, using all network interfaces available on the host computer.

B.2 Resource deployment

At least one resource is required to run CTMS and can be deployed as follows:

```
java -jar CTMS.jar -r <host:port> resource
```

Listing B.2: Resource creation

It is necessary to provide the resource with the network name of the computer hosting SDS and the port number on which the service accepts requests.

Appendix C

Paper - Cloud Computing Concept for Intelligent Transportation Systems

Cloud Computing Concept for Intelligent Transportation Systems

Paweł Jaworski, Tim Edwards, Jonathan Moore, Keith Burnham

Abstract — **In this paper a cloud computing based urban traffic control system is proposed. Its goals are to increase road throughput and optimise the traffic control for increased safety of the participants, reduced fuel consumption and carbon emissions. The urban vehicle control scenario assumes that the speed of each vehicle in the controlled area is set by an off-board control unit that supervises each traffic intersection. The software component responsible for that is called an Intersection Control Service (ICS). From the system's point of view, the vehicles are treated as cloud services and are discovered and invoked using a cloud computing methodology. Geographical multicast addressing is used to target all vehicles in the specified areas. ICSs are part of a city/region wide cloud system that coordinates flow of traffic between intersections. The system's optimisation objective is carried out on several planning planes simultaneously, the lowest being local to a single intersection and the highest being an entire city or region level. The ICS gathers traffic data from various sensors around the intersection, and from the vehicles themselves, creating a dynamic situation map which can be used to assess the road situation and perform short term predictions for vehicle control purposes.**

This item has been removed due to third party copyright.
The unabridged version of the thesis can be viewed at the
Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

Appendix D

Paper - Distributed Traffic Flow Optimisation and Control for Intelligent Transportation Systems

Distributed Traffic Flow Optimisation and Control for Intelligent Transportation Systems

Paweł Jaworski*. Tim Edwards.*
Keith Burnham. Olivier Haas.****

*MIRA Ltd., Nuneaton, CV10 0TU, UK
(e-mail: pawel.jaworski@mira.co.uk).
**Coventry University, Coventry, UK

Abstract: This paper describes an approach to traffic control that takes advantage of modern ITS capabilities such as Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communication, and advanced vehicle sensing techniques. The aim is to efficiently manage traffic flow in a given area. Vehicles equipped with such means of communication can actively participate in the traffic control process by adjusting their speed according to the intersection control algorithm's (ICA) demands, thus enforcing desired behaviour upon other traffic participants.

Keywords: traffic control, distributed computing, intelligent transportation systems

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

Appendix E

Paper - Microscopic Traffic Simulation Tool for Intelligent Transportation Systems

Microscopic Traffic Simulation Tool for Intelligent Transportation Systems

Paweł Jaworski, Tim Edwards, Keith Burnham, Olivier Haas

***Abstract* — This paper describes a microscopic traffic simulation tool for assessing the performance of ITS traffic and vehicle control systems. The scope of simulation is very broad. The traffic network is simulated in microscopic scale with nanoscopic components being available as well. The vehicles can be simulated down to basic physical properties including throttle and brake settings, fuel consumption and others. This allows for in-depth behavioural analysis of both individual vehicles and the whole traffic flow as a result of use of different traffic management systems. The simulator also allows investigating inter-vehicle interactions including platooning behaviour and its consequences such as the string stability problem. The simulator has been designed to be modular and easily extensible to allow for new functionality in future.**

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

Appendix F

Paper - A Network Assisted Vehicle for ADAS and ITS testing

The author contributed to this paper by designing and implementing the data base system and longitudinal control algorithm for the vehicle described in the paper.

A Network Assisted Vehicle for ADAS and ITS testing

Tim Edwards, Jonathan Moore, Maria Loukadaki and Pawel Jaworski

***Abstract*—Rapid developments in technologies such as embedded devices, with increased processing capability, and sensor systems, with high accuracy and fast response, are enabling a wide range of new road traffic applications. These new systems and services promise significant improvements in areas such road safety, fuel economy and congestion management. In many cases the commercial deployment of these Advanced Driver Assistance Systems (ADAS), or Intelligent Transportation Systems (ITS), is no longer limited by just technical or economic constraints. There are ethical, legal or certification issues that need to be resolved for wide scale deployment.**

In this paper we present the Network Assisted Vehicle (NAV) concept. NAV is a prototype semi-autonomous vehicle with a modular design that can be adapted to support new ADAS and ITS test standards. Vehicle costs are minimized by making full use of existing OEM systems on the vehicle and the known conditions of a controlled test environment (innovITS ADVANCE City Circuit test facility).

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

Appendix G

Paper - Autonomous longitudinal control for a Network Assisted Vehicle

The author contributed to this paper by designing and implementing the data base system and longitudinal control algorithm for the vehicle described in the paper. Section 3 of the paper was written by the author of this thesis.

Autonomous longitudinal control for a Network Assisted Vehicle

Tim Edwards, Pawel Jaworski and Maria Loukadaki

MIRA Ltd, Watling Street,
Nuneaton, Warwickshire, CV10 0TU, UNITED KINGDOM
Phone: +44 24 7635 5484
Fax: +44 24 7635 8484
E-mail: tim.edwards@mira.co.uk

The Network Assisted Vehicle is a semi-autonomous vehicle that is tightly integrated with the innovITS ADVANCE City Circuit test facility. The vehicle incorporates a low cost speed control system that utilizes a pre-existing adaptive cruise control system. A radio interface allows synchronization with an off-board control facility that can adjust the vehicle speed demand around other events on the track. This paper describes the design and evaluation of the both the low level on-board speed control system and the higher level off-board algorithm.

Topics/Intelligent Transportation Systems, Semi-autonomous vehicles, Active safety testing

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.

This item has been removed due to third party copyright. The unabridged version of the thesis can be viewed at the Lanchester Library, Coventry University.